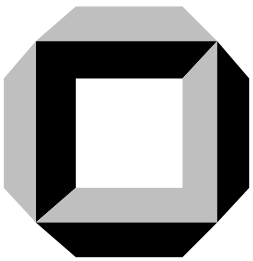
INSTITUT FÜR WIRTSCHAFTSTHEORIE
UND OPERATIONS RESEARCH
UNIVERSITÄT KARLSRUHE

Projects with Minimal and Maximal Time Lags: Construction of Activity-on-Node Networks and Applications

Klaus Neumann
Christoph Schwindt

Report WIOR-447



TECHNICAL REPORT

Kaiserstraße 12 · D - 76128 Karlsruhe · Germany

Projects with Minimal and Maximal Time Lags: Construction of Activity-on-Node Networks and Applications

Klaus Neumann
Christoph Schwindt

Report WIOR-447

June 1995

Abstract

Maximal time lags between activities of a project play an important role in practice in addition to minimal ones. However, maximal time lags have been discussed very rarely in literature thus far. Projects with minimal and maximal time lags can be modeled by cyclic activity-on-node networks. The production process for make-to-order production can be represented by a multi-project network, where the individual operations of the jobs correspond to the nodes of the network. For different product structures, careful consideration is given to the modelling of a nondelay performance of overlapping operations by appropriately establishing minimal and maximal time lags. Additional applications of maximal time lags (for example, to represent prescribed milestones, due dates, time windows, or time-varying resource-requirements for activities) are discussed as well.

Key Words

Project planning and control, resource-constrained project scheduling, activity-on-node networks, maximal time lags, make-to-order production.

Contents

1. Introduction

In the late 1950s, methods for planning and scheduling projects were developed based on network models. Two different types of *project networks* (also called *activity networks*) were introduced. If the activities of the project are assigned to the arcs of a network (and the durations of the activities correspond to the arc weights), we speak of on *activity-on-arc network* (or briefly *A-on-A network*). If the activities of the project are assigned to the nodes of a network (and the arc weights represent minimal or maximal time lags between activities), we speak of an *activity-on-node network* (or briefly *A-on-A network*).

A-on-A networks have been widely discussed in literature and used in practice (cf. Elmaghraby 1977, Neumann 1975, Neumann & Morlock 1993, Wiest & Levy 1977). The so-called *Critical Path Method* (abbreviated *CPM*), the method of project planning and control which is most often employed in applications, is based on A-on-A networks. On the other hand, A-on-N networks and the *Metra-Potential Method* (abbreviated *MPM*) based on A-on-N networks have been found very rarely in literature and practice until recently (cf. Elmaghraby & Kamburowski 1992, Neumann 1975, Neumann & Morlock 1993, Roy 1964). The latter fact is amazing because A-on-A networks have several disadvantages in comparison with A-on-N networks, for example:

(a) To model all the precedence relations among the activities of a project in an A-on-A network, a large number of *dummy activities* have to be introduced in general, which cannot be done in a unique way and is a frequent source of error. In fact, the problem of finding an A-on-A network with minimum number of dummy activities which corresponds to a given project is NP-hard (Garey & Johnson 1979).

(b) Only *minimal* time lags between different activities can be modeled by an A-on-A network.

This paper gives an introduction to A-on-N networks, which clearly shows the advantages of A-on-N networks over A-on-A networks. In particular, the paper avoids several careless mistakes in constructing A-on-N networks which are found in literature, and it stresses the importance of maximal time lags in practice. Section 2 deals with minimal and maximal time lags between activities.The construction of an A-on-N network is discussed in Section 3. Sections 4 and 5 are concerned with applications. Single-item and small-batch production generally represent a *make-to-order production*. To carry out a production order can be viewed as a project. How to model the entire production process, which comprises a large number of production orders, by a multi-project network is shown in Section 4. Also, the case of operation overlapping is discussed. Section 5 presents several additional examples from practice that show the importance of maximal time lags in addition to minimal ones.

2. Minimal and Maximal Time Lags between Activities

Suppose that the project in question consists of n activities numbered 1 to n, where each activity is to be carried out without interruption. In addition the fictitious activities 0 and $n+1$ are introduced, which represent the beginning and completion of the project, respectively.

Let $D_j \geq 0$ be the *duration* of activity j where $D_0 = D_{n+1} = 0$. Moreover, let ST_j be the *start time* of activity j where we put $ST_0 := 0$. Then the *project duration* equals

$$ST_{n+1} = \max_{j=0,1,\ldots,n+1} \left(ST_j + D_j \right).$$

An activity $j \neq 0$ is called an *initial activity* of the project if there is no activity $l \neq j$ that has to be begun before starting activity j. An activity $j \neq n+1$ is called a *terminal activity* of the project if there is no activity $l \neq j$ which is begun after completing activity j.

Next, we introduce minimal and maximal time lags between the start of two different activities. A *minimal time lag* $T_{jl}^{min} \geq 0$ between the start of two activities j and l says that

(2.1) $$ST_l - ST_j \geq T_{jl}^{min}.$$

If there is a minimal time lag $T_{jl}^{min} \geq 0$ between activities j and l, then l is said to *follow* j. In case that activity l can be begun before the completion of activity j, that is, activities j and l "overlap", we have $T_{jl}^{min} < D_j$.

Remark 1. If two activities j and l can start at the same point in time, we set either $T_{jl}^{min} := 0$ or $T_{lj}^{min} := 0$.

We consider two particular minimal time lags. Sometimes *release dates* or *ready times* are prescribed for some activities. Let $r_j \geq 0$ be a given release date of activity j, that is, activity j is available for performing at time r_j. Then $ST_j - ST_0 \geq T_{0j}^{min}$ where $T_{0j}^{min} = r_j$. In particular, for each initial activity j of the project, a minimal time lag T_{0j}^{min} is given. If initial activity j can be started at the beginning of the project, then $T_{0j}^{min} = 0$.

If activity j has to be completed a prescribed period of time $T_{j,n+1}$ before the termination of the project at the latest, then $T_{j,n+1}^{min} := T_{j,n+1} + D_j$ represents a minimal time lag: $ST_{n+1} - ST_j \geq T_{j,n+1}^{min}$. In particular, for each terminal activity j of the project, a minimal time lag $T_{j,n+1}^{min}$ is given. If terminal activity j only needs to be completed at the termination of the project, then $T_{j,n+1}^{min} = D_j$.

Remark 2. If for activity j and for all activities l that follow j it holds that $T_{jl}^{min} + D_l < D_j$, then a minimal time lag $T_{j,n+1}^{min} := D_j$ has to be introduced.

Remark 2 ensures that activity j will be completed at the termination of the project.

A *maximal time lag* $T_{jl}^{max} \geq 0$ between the start of two activities j and l says that

(2.2) $$ST_l - ST_j \leq T_{jl}^{max}.$$

Remark 3. The assumption that minimal and maximal time lags are to be nonnegative does not mean any loss of generality. Suppose that there is a minimal time lag $T_{jl}^{min} < 0$ between activities j and l. Putting $T_{lj}^{max} := -T_{jl}^{min} > 0$, (2.1) becomes

$$ST_j - ST_l \leq T_{lj}^{max}$$

that is, we have a positive maximal time lag T_{lj}^{max} between activities l and j. Similarly, negative maximal time lags can be replaced by positive minimal time lags.

Assumption 1. If there is a maximal time lag T_{jl}^{max} between two activities j and l, there exist a sequence of activities $j_0 = j, j_1, \ldots, j_{m-1}, j_m = l$ and minimal time lags $T_{j_0 j_1}^{min}, \ldots, T_{j_{m-1} j_m}^{min}$. Moreover, it has to hold that

$$T_{j_0 j_1}^{min} + \ldots + T_{j_{m-1} j_m}^{min} \leq T_{jl}^{max}. \tag{2.3}$$

Obviously, in practical projects, a maximal time lag T_{jl}^{max} only makes sense if Assumption 1 is satisfied. In Section 3 we will see that if Assumption 1 is violated, there is no feasible time schedule of the project.

Remark 4. If two activities j and l have to start at the same point in time, we set either $T_{jl}^{min} := T_{jl}^{max} := 0$ or $T_{lj}^{min} := T_{lj}^{max} := 0$.

We consider two particular maximal time lags. Sometimes *deadlines* are prescribed for some activities. Let $d_j \geq D_j$ be a given deadline for activity j, that is, activity j has to be completed by time d_j. Then $ST_j - ST_0 \leq T_{0j}^{max}$ where $T_{0j}^{max} = d_j \geq D_j$. If activity j can be begun a prescribed period of time $T_{j,n+1}^{max}$ before the termination of the project at the earliest, then $T_{j,n+1}^{max}$ represents a maximal time lag: $ST_{n+1} - ST_j \leq T_{j,n+1}^{max}$.

Some time constraints that occur in practice frequently can be expressed in terms of minimal and maximal time lags, for example:

(a) Activity l has to be carried out after activity j without any delay, that is, $ST_l - ST_j = D_j$. This constraint can be ensured by introducing minimal and maximal time lags such that $T_{jl}^{min} = T_{jl}^{max} = D_j$.

(b) If activity j has to be begun at the point in time T_{0j}^{min} exactly, we introduce minimal and maximal time lags such that $T_{0j}^{min} = T_{0j}^{max}$.

Since the minimal and maximal time lags introduced above connect the start times of two activites, they are also called *start-to-start* time lags. In a similar manner, we may introduce *finish-to-finish*, *start-to-finish*, and *finish-to-start* time lags. Those four different time lags can easily be converted into one another. As an example, consider the conversion of finish-to-start time lags into start-to-start time lags and vice versa. Let $FT_j := ST_j + D_j$ be the *finish time* of activity *j*. Moreover, let ${}^{SS}T_{jl}^{min}$ and ${}^{SS}T_{jl}^{max}$ (instead of T_{jl}^{min} and T_{jl}^{max} as above) be the minimal and maximal start-to-start time lags, respectively, and let ${}^{FS}T_{jl}^{min}$ and ${}^{FS}T_{jl}^{max}$ be the minimal and maximal finish-to-start time lags, respectively. Then

$$ST_l - FT_j \geq {}^{FS}T_{jl}^{min}$$
$$ST_l - FT_j \leq {}^{FS}T_{jl}^{max}$$

where

$${}^{FS}T_{jl}^{min} = {}^{SS}T_{jl}^{min} - D_j$$
$${}^{FS}T_{jl}^{max} = {}^{SS}T_{jl}^{max} - D_j$$

(compare (2.1) and (2.2)).

3. Construction and Properties of an Activity-on-Node Network

We assign the nodes 0, 1,..., *n*+1 of a network to the activities 0, 1,..., *n*+1 of the project in question (and identify the activities with the nodes). If there is a minimal time lag T_{jl}^{min} between activities *j* and *l*, we introduce an arc $<j,l>$ with weight $b_{jl} := T_{jl}^{min}$. If there is a maximal time lag T_{jl}^{max} between activities *j* and *l*, we introduce a *backward arc* $<l,j>$ with weight $b_{lj} := -T_{jl}^{max}$. Let $V=\{0, 1,..., n+1\}$ be the node set and *E* be the arc set of the resulting A-on-N network. The time constraints (2.1) and (2.2) can then be summarized as

(3.1) $ST_l - ST_j \geq b_{jl}$ for all $< j,l > \in E$

We state some *properties of an A-on-N network* where for the basic concepts from the theory of graphs and networks, we refer to Neumann & Morlock (1993). By Remarks 1 and 4, an A-on-N network does not contain parallel arcs. In general, there are positive arc weights (corresponding to minimal time lags) as well as negative arc weights (corresponding to maximal time lags) in an A-on-N network. By Assumption 1, each backward arc (with nonpositive weight) belongs to a cycle of the network. Inequality (2.3) guaranties that there are no cycles of positive length. Since an A-on-N network corresponds to a real project, the network is weakly connected. Moreover, each node *j* is reachable from node 0 (beginning of the project), and from each node *j*, node *n*+1 (termination of the project) is reachable. By the above construction, the A-on-N network assigned to a project with given minimal and maximal time lags is uniquely determined aside from the orientation of *null cycles*, that is, cycles all of whose arcs have weight 0.

A sequence of start times $(ST_0, ST_1, ..., ST_{n+1})$ where $ST_0 = 0$ and $ST_j \geq 0$ for $j = 1, ..., n+1$ is called a *(time) schedule* of a project or a respective A-on-N network. A schedule is said to be *feasible* if it satisfies inequality (3.1). A feasible schedule that minimizes the project duration ST_{n+1} is called *(time-) optimal*. Let D^* be the *minimum project duration*.

An *earliest schedule* $(EST_0, EST_1, ..., EST_{n+1})$ is a feasible schedule with $EST_j \leq ST_j$ (j=0, 1,...,n+1) for all feasible schedules $(ST_0, ST_1, ..., ST_{n+1})$. Analogously, a *latest schedule* $(LST_0, LST_1, ..., LST_{n+1})$ is a feasible schedule with $LST_{n+1} = D^*$ and $LST_j \geq ST_j$ (j=0, 1,...,n+1) for all feasible schedules $(ST_0, ST_1, ..., ST_{n+1})$. Note that $EST_0 = LST_0 = 0$ and $EST_{n+1} = D^*$.

It is well-known that the *earliest start time* EST_j of activity j equals the length of a longest path from node 0 to node j in the respective A-on-N network. Similarly, for the *latest start time* LST_j, $D^* - LST_j$ equals the length of a longest path from node j to node n+1. If a deadline $d \geq D^*$ for the completion of the project is prescribed, then D^* has to be replaced by d for latest start times LST_j. The *temporal analysis* of a project or a corresponding A-on-N network consists of the computation of the earliest and latest start times EST_j and LST_j (j=0, 1,...,n+1), which can be done in $O(|V|\,|E|)$ time by a so-called label-correcting algorithm (see Neumann & Morlock 1993).

Let us now drop the second part of Assumption 1 and establish the first part as

Convention 1. If there is a maximal time lag T_{jl}^{max} between two activities j and l, there exist a sequence of activities $j_0 = j, j_1, ..., j_m = l$ and minimal time lags $T_{j_0 j_1}^{min}, ..., T_{j_{m-1} j_m}^{min}$.

Note that Convention 1 ensures that each backward arc belongs to a cycle, however, it does not exclude cycles of positive length. Convention 1 means no loss of generality because it can always be satisfied, if necessary, by introducing a dummy activity or respectively node and auxiliary arcs (compare Brinkmann 1992).

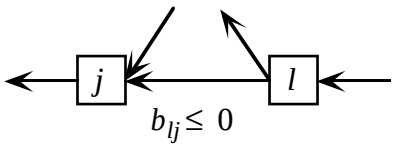


Fig. 1

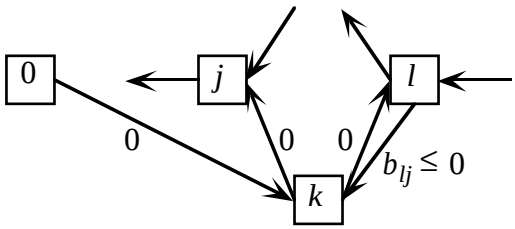


Fig. 2

Consider the example shown in Figure 1 where <*l*,*j*> is a backward arc and it is supposed that there is no path from node *j* to node *l*. Figure 1 can be replaced by Figure 2 where *k* is a dummy node or respectively activity of duration 0 and <0,*k*>, <*k*,*j*>, and <*k*,*l*> are auxiliary arcs with weight 0. Figure 1 says that

(3.2) $ST_l - ST_j \leq T_{jl}^{max} = -b_{lj}$.

By Figure 2

(3.3) $ST_j - ST_k \geq 0$ or $ST_j \geq ST_k$

and

$$0 \leq ST_l - ST_k \leq -b_{lj}$$

which implies

(3.4) $ST_k \geq ST_l + b_{lj}$.

From (3.3) and (3.4) it follows that

$$ST_j \geq ST_l + b_{lj}$$

that is, (3.2) is satisfied. Moreover, the length of longest paths from node 0 to nodes *j* and *l*, respectively, and the longest path lengths from nodes *j* and *l*, respectively, to node *n*+1 in the original A-on-N network coincide with the corresponding path lengths in the network obtained by replacing Figure 1 with Figure 2. As a consequence, the temporal analysis in both networks provides the same results.

Finally, we state one of the main results on A-on-N networks.

Theorem 1. There exists a feasible schedule exactly if the respective A-on-N network does not contain any cycle of positive length.

For the proof we refer to Neumann (1975) or Bartusch et al. (1988).

4. Multi-Project Networks in Make-to-Order Production

4.1 Basic Concepts

In Section 3, a *feasible* schedule has been defined as a sequence of start times which satisfies inequality (3.1). An implicit assumption of this definition and the corresponding temporal analysis is the availability of unlimited capacity of resources needed for the execution of activities. In reality, however, resources are always scarce.

In the following, we will assume that *K* renewable resources *i* (*i*=1,...,*K*) are required for carrying out the project. A resource *i* is called *renewable* if its (limited) capacity R_i is available in each period independently of its utilization in previous periods. Let $0 \leq r_{ij} \leq R_i$ be

the period capacity of resource i required for the execution of activity j (j = 1,...,n; i = 1,...,K). All parameters (minimal and maximal time lags, per period capacities of resources, per period usage of resources, and activity durations) are supposed to be integer-valued. We assume that each activity is carried out without interruption (nonpreemptive processing).

Let $V(t) := \left\{ j \in \{1,\dots,n\} \middle| t - D_j < ST_j \le t \right\}$ be the set of activities in execution at time t or in time interval [t,t+1[, respectively. Let T be an upper bound on the project duration. The Resource-Constrained Project Scheduling Problem with minimal and maximal time lags (RCPSP / max) for the A-on-A network N of Section 3 can be stated as follows:

$$
\text{(RCPSP / max)} \qquad \begin{cases} \min & ST_{n+1} & \\ \text{s.t.} & ST_l - ST_j \ge b_{jl} & (< j,l > \in E) \\ & \sum_{j \in V(t)} r_{ij} \le R_i & (i = 1,\dots,K; t = 0,\dots,T-1) \\ & ST_0 = 0 & \\ & ST_j \in Z_+ & (j = 1,\dots,n) \\ & \text{Activity splitting is not allowed} & \end{cases}
$$

For a formulation of RCPSP / max as a linear optimization problem with binary variables we refer to Franck and Schwindt (1995). The decision problem whether there is a feasible solution of a given RCPSP / max (feasibility problem) is in general NP-hard (see Bartusch et al. 1988).

The Resource-Constrained Project Scheduling Problem without maximal time lags (RCPSP) has been treated by numerous authors (Christofides et al. 1987, Demeulemeester & Herroelen 1992, Stinson et al. 1978, Talbot & Patterson 1978). Exact algorithms for RCPSP / max can be found in Bartusch (1983) and Bartusch et al. (1988), for heuristic procedures we refer to Brinkmann & Neumann (1994), Neumann & Zhan (1995), and Zhan (1994).

One of the most promising fields of application for resource-constrained project scheduling is the production scheduling in make-to-order production. A production system is classified as make-to-order if all products are manufactured only in response to customer orders, that is, no inventories are built up for future sale.

A well-known concept for production planning in manufacturing is Materials Requirements Planning (MRP). MRP systems are computerized data processing systems to schedule production and control the level of inventory for components. MRP consists of four phases: determination of gross requirements of final products, subassemblies, and components (bill of materials explosion), determination of net requirements (based on the gross requirements, scheduled receipts, and inventory), lot-sizing, and time phasing (taking into account production lead times). MRP analysis provides order releases for final products, subassemblies, and components which have to be processed on a given set of machines. For details we refer to Chase & Aquilano (1989), Evans et al. (1990), Heizer & Render (1993), or Nahmias (1993).

In contrast to assemble-to-order systems, in make-to-order systems even components and subassemblies are manufactured only if they are required for the production of a customer-ordered final product. Therefore, phases 2 and 3 of MRP (determination of net requirements and lot-sizing) are skipped in make-to-order production.

Since MRP does not explicitly take resource capacities into consideration, usually revisions in the scheduled order releases will be necessary, often resulting in large flow times of the ordered products. Recently, a new (capacity-oriented) MRP concept has been presented by Drexl et al. (1994), which tries to overcome this drawback.

The separation of material and capacity requirements planning can be avoided by using algorithms for resource-constrained scheduling. On an aggregate level, we will determine *milestones* (latest *start* times) for the production of components and subassemblies by a resource-unconstrained temporal analysis in an aggregate network. The detailed scheduling of all operations which need to be performed for the manufacturing of customer orders will be based on a multi-project network. Together with the resource data, this network defines an instance of RCPSP / max corresponding to an appropriate formulation of the basic make-to-order production scheduling problem.

In this section, we show how to transform a given production scheduling problem into a problem of type RCPSP / max which allows for modelling both overlapping operations and delivery dates for customer orders.

The following input data are supposed to be available:

(1) *Customer orders.* A customer order (cf. Figure 3) consists of
- several *products* $j \in J$
- *demands* (*order quantities*) x_j^d, $(j \in J)$
- *delivery dates* (*deadlines*) d_j, $(j \in J)$

For convenience, we assume that each customer-ordered product belongs to exactly one customer order. If there is more than one order of product j, we let x_j^d be the sum of all order quantities of product j and the deadline d_j be the minimum of all deadlines belonging to product j.

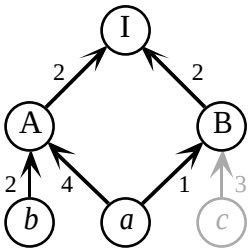
Customer	Product j	Part #	Order quantity x_j^d	Delivery date d_j
X	product I	23	2	200
X	product B	13	1	120
Y	product *b*	11	3	130
X	product II	37	1	250

Fig. 3: Customer orders

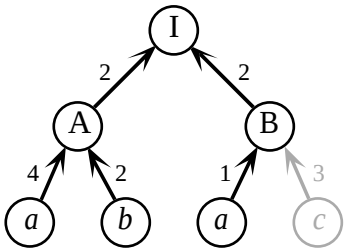
(2) *Bills of materials*, *gozinto graphs*, or *product trees* (with the *input coefficient* a_{jl} denoting the number of units of product j which are directly built in one unit of product l). Usually, parts can be divided into components, subassemblies, and final products (cf. Figure 4, where *c* is a purchased component).

Bill of Materials	Part #
1 final product I	23
2 subassemblies A	17
2 components *b*	11
4 components *a*	7
2 subassemblies B	13
1 component *a*	7
3 components *c*	3

Bill of materials



Gozinto graph



Product tree

Fig. 4: Bill of materials, gozinto graph, and product tree

In the following, x_j will denote the number of units of product j required for the production of all customer-ordered products. Since in manufacturing gozinto graphs do not contain cycles, x_j can easily be determined by evaluating the following expression for each product j in the order of nondecreasing levels (the level of product j is defined to be the maximal number of arcs of a path in the gozinto graph from j to a final product):

$$x_j := x_j^d + \sum_{l \in \left\{ l' \,\middle|\, a_{jl'} > 0 \right\}} a_{jl} x_l \quad (j \in J).$$

For the example given by Figures 3 and 4 we obtain: $x_{\mathrm{I}} = 2$, $x_{\mathrm{A}} = 4$, $x_{\mathrm{B}} = 5$, $x_a = 21$, $x_b = 11$, and $x_c = 15$ (without the units needed for the assembly of product II).

Since lot sizing is dropped in make-to-order production, x_j corresponds to the processed *lot size*. In what follows, a *job* j will stand for the manufacturing of *all* x_j *units* of *product* j.

(3) *Delivery dates* r_j of *purchased components*. If product j has to be purchased, r_j denotes that time where all x_j required units of product j become available. If the x_j units of product j are in stock, we set $r_j := 0$. Figure 5 shows the delivery date of a purchased product c, which is a component of both final products I and II.

Supplier	Component j	Part #	Order quantity x_j	Delivery date r_j
Z	component c	3	20	15

Fig. 5: Purchased components

(4) The number m_i of available *machines* of type M_i (i=1,...,K). The machines correspond to the renewable resources in RCPSP / max.

(5) Order-independent *schedules of (job) operations* which contain information about
- the set M_j of machines M_i on which product j is to be processed,
- the sequence of activities or *operations* of job j determined by technological and organizational restrictions, where each operation corresponds to the processing of job j on exactly one of the machines from M_j ,
- the *setup time* ϑ_{ij} and the *processing time* p_{ij} of a single unit of job j on machine M_i.

By an operation (i,j) we mean the setup of machine M_i and the processing of all x_j units of product j on machine M_i. The duration D_{ij} of (i,j) equals $\vartheta_{ij} + x_j p_{ij}$. An operation corresponds to an activity in problem RCPSP / max.

Figure 6 shows the schedule of operations for final product I given by Figure 4.

Product j	Machines M_i	Processing time p_{ij}	Setup time ϑ_{ij}
I	M_1	5	3
	M_2	3	2
A	M_4	7	1
	M_3	2	2
	M_1	3	1
B	M_2	2	1
	M_4	2	3
a	M_4	1	2
b	M_1	1	1
	M_4	2	2
	M_2	1	2

Fig. 6: Schedule of operations

(6) *Transportation lot size* q_{ik}^j of product j from machine M_i to M_k. If after the processing on machine M_i product j is built in a subassembly or final product l, q_{ik}^j / a_{jl} is assumed to be integer-valued.

(7) *Transfer time* t_{ik}^j of q_{ik}^j units of product j from machine M_i to M_k.

(8) *Minimum number* f_{ik}^j of units of product j which have to be finished on machine M_i before some transfer from M_i to M_k can be started (for the first time).

Note that we have to distinguish between three different kinds of times:

- processing time p_{ij}: during processing time p_{ij}, both machine M_i and exactly one unit of product j are not available for other tasks.
- setup time ϑ_{ij}: machine M_i is not available during the setup time, whereas units of product j can still be processed on preceding machines. In real-life applications, the setup often cannot be done without having available one of the units to be processed. In this case, the corresponding part of the setup time has to be interpreted as part of the total processing time of the x_j units.
- transfer time t_{ik}^j: obviously, a unit transferred from one machine to another is not available for processing. Machines are not affected by part transfer. Transportation times as well as waiting times (due to technological reasons, for example, the time for drying lacquer or glue) can be considered to be transfer times.

In the following four Subsections 4.2 to 4.5, we will propose an approach for modelling scheduling problems in make-to-order production using appropriate A-on-N networks.

Subsection 4.2 deals with the construction of a simple acyclic A-on-N network which is used for the computation of milestones. Here resource constraints will not yet be taken into consideration.

The fixing of minimal and maximal time lags between the start of different operations will be treated in Subsection 4.3. In the case of so-called *repeat parts* (parts which are components of several subassemblies or final products), we have to determine a sequence in which the units of repeat parts have to be allotted to the products in which they are built. This sequence generally has a strong impact on the completion times of the ordered products.

In Subsection 4.4, it will be shown how to use minimal and maximal time lags together with the bills of materials, schedules of operations, release dates of purchased products, and deadlines for customer-ordered products to construct a cyclic A-on-N network, where each operation corresponds to an activity.

The network constructed will be used in Subsection 4.5 for establishing a project scheduling problem of type RCPSP / max corresponding to the given make-to-order production scheduling problem. This problem can be solved approximately by one of the heuristic algorithms from literature mentioned above.

Figure 7 shows the individual phases of the modelling process.

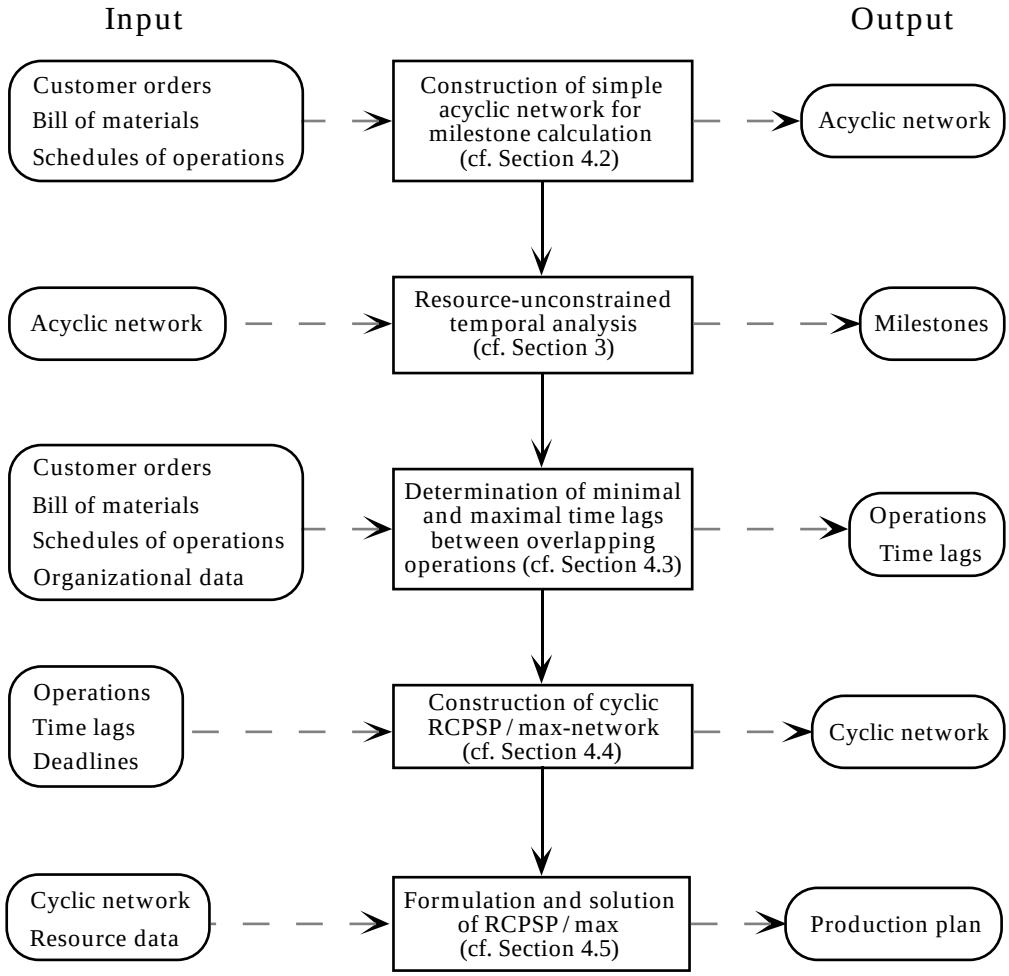


Fig. 7: Flow chart for modelling a make-to-order production scheduling problem

4.2 Milestones for the Start Times of Components, Subassemblies, and Final Products

Milestones can be used on an aggregate level to answer questions such as:

- Can due dates be met? If not, which date can be promised to the customer?
- Manufacturing of which products seems to be (time) critical?
- Will it be necessary to schedule additional shifts? How many workers will be required and in which period of time?

For the computation of milestones, we perform a temporal analysis in acyclic A-on-N networks (cf. Section 3). An activity j (or the corresponding node in the network, respectively) will represent the production of all x_j units of product j.

Units of intermediate (that is, non-final) products which have been customer-ordered will not be needed for further processing. Hence, we have to distinguish between the manufacturing of units which are directly customer-ordered and the manufacturing of units which are components or subassemblies of customer-ordered products.

The following Algorithm 1 constructs an acyclic A-on-N network for a single final product. Since the temporal analysis does not take care of the dependencies between the manufacturing of several final products which are due to the scarce resources, we perform a separate temporal analysis for each final product (which is advantageous as far as the computational effort is concerned).

Algorithm 1. *Construction of an acyclic A-on-N network N_l for the determination of milestones for final product l*

Start with the network structure of the gozinto graph of product l. Remove all nodes (and incident arcs) which correspond to purchased components. Weight each remaining arc $< j_1, j_2 >$ with $\sum_{M_i \in M_j} \left(\vartheta_{ij_1} + a_{j_1 j_2} x_{j_2} p_{ij_1} \right)$. The weight corresponds to the time needed for the manufacturing of all units of product j_1 that are required for the assembly of all x_{j_2} units of product j_2.

Introduce a supersource α and arcs from α to all nodes representing components j, and weight the arcs $<\alpha,j>$ with 0. Add further arcs from α to the subassemblies j in which purchased components are directly built. Weight these arcs $<\alpha,j>$ with delivery date $r_{j'}$, where j' is the purchased component with maximal delivery date built in j.

Connect each customer-ordered product j with an artificial sink ω_j and weight the corresponding arc $<j,\omega_j>$ with $\sum_{M_i \in M_j} \left(\vartheta_{ij} + x_j^d p_{ij} \right)$. The latter weight corresponds to the time needed for the manufacturing of all ordered units of product j. ❑

Figure 8 shows the resulting A-on-N network N_I for product I of Figure 4.

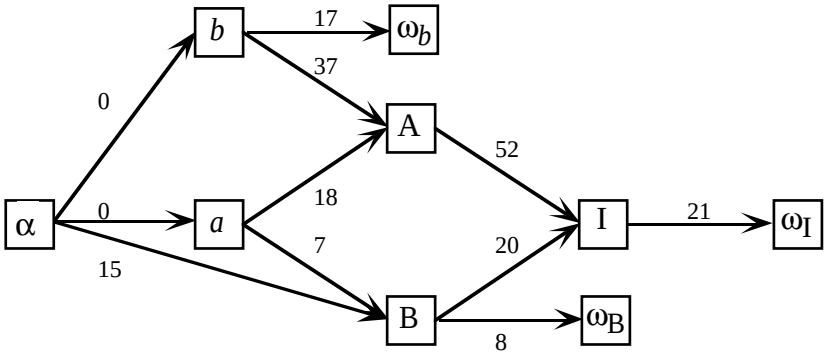


Fig. 8: Network for the determination of milestones

Let FP_j be the set which contains product j and all final products l in which units of product j are built. Let $L_l(j_1,j_2)$ be the length of a longest path from node j_1 to node j_2 in the network N_l of product l. If a product j has not been customer-ordered, we set $d_j := \infty$. The milestone MS_j for the start time of product j can be determined as follows:

$$MS_j := \min_{l \in FP_j} \left\{ \min \left\{ d_l - L_l(j,\omega_l); d_j - \sum_{M_i \in M_j} \left(\vartheta_{ij} + x_j^d p_{ij} \right) \right\} \right\}.$$

Since we are only interested in longest paths from nodes j to sinks ω_l, the temporal analysis can be restricted to the calculation of latest start times for products j.

For the example given by Figure 8 we obtain $MS_{\mathrm{I}} = 179$, $MS_{\mathrm{B}} = 112$, $MS_{\mathrm{A}} = 127$, $MS_b = 90$, $MS_a = 105$.

Remark 5. Since we do not take resource constraints into consideration, the milestones tend to be too large, that is, the corresponding activities will be started too late. This effect, however, is lessened by the fact that we have not considered the possibility of overlapping operations yet, which will be discussed in Subsection 4.3.

4.3 Minimal and Maximal Time Lags between Overlapping Operations

If $x_j > 1$, the time lag between the start of two successive operations (i,j) and (k,j) may be less than the duration D_{ij} of operation (i,j). To allow such an *overlapping* of successive operations, units belonging to job j have to be transferred to M_k before the completion of the preceding operation (i,j). Of course, we may transfer units of j from M_i to M_k as soon as they have been processed on M_i. In that case, however, if $p_{ij} > p_{kj}$, the processing of operation (k,j) will be interrupted after the processing of each unit of product j. Next, we will show how to use minimal and maximal time lags such that the processing of an operation will not be interrupted, that is, no idle times will occur between the processing of two successive units of one and the same product on a given machine.

Günther (1992) employs *minimal* start-to-start and finish-to-finish time lags for some of the cases treated in what follows. As we will see, overlapping operations may require the use of maximal time lags, too. We will only use start-to-start time lags. Different types of time lags can easily be converted into start-to-start time lags as shown in Section 2.

If there are repeat parts, we have to decide on the sequence in which the units of a repeat part have to be allotted to products in which they will be built. The approach of Günther (1992) for the (implicit) scheduling of repeat parts can only be used for a temporal analysis relaxing all resource constraints. Therefore, we will propose new algorithms for the sequencing of the allotment of repeat parts j (*allotment sequencing problem ASP(j)*) which can be used in case of scarce resources.

(a) Linear product structure

The structure of a product is called linear, if each node of the corresponding gozinto graph has at most one (immediate) predecessor and at most one (immediate) successor. No assembly operations are to be performed on such a product.

We first consider the case $t_{ik}^{j} = 0$, $q_{ik}^{j} = 1$, and $f_{ik}^{j} = 1$ for all $j \in J$, $M_i, M_k \in M_j$, with (k,j) being the operation succeeding operation (i,j) in the schedule of operations for product j. We have to distinguish between two subcases:

(*i*) $p_{ij} \le p_{kj}$

Each unit of product j completed on machine M_i can immediately be transferred to machine M_k. The setup of M_k can be accomplished before the arrival of the first unit of j. For the time lag $T_{ij,kj}$ between the start of operations (i,j) and (k,j) we obtain (cf. Figure 9)

(4.1) $$T_{ij,kj} = \vartheta_{ij} + p_{ij} - \vartheta_{kj}.$$

If $\vartheta_{kj} \le \vartheta_{ij} + p_{ij}$, then $T_{ij,kj}$ is nonnegative and we introduce a minimal time lag $T_{ij,kj}^{min} := T_{ij,kj}$. Otherwise, $T_{ij,kj}$ becomes negative and we introduce a maximal time lag $T_{kj,ij}^{max} := -T_{ij,kj} > 0$ (compare Section 2). All time lags $T_{\cdot,\cdot}$ occurring in present Section 4 can either be nonnegative or negative. Accordingly, we have to introduce minimal or respectively maximal time lags between the start of the corresponding operations.

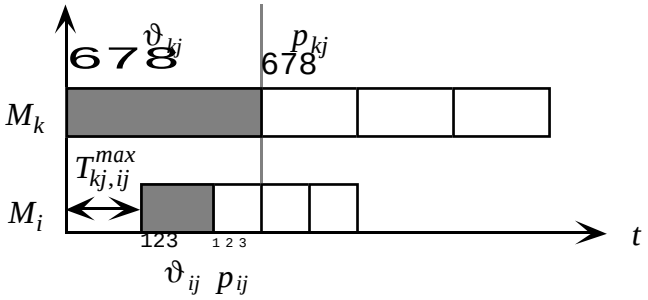


Fig. 9: Overlapping operations with $p_{ij} \le p_{kj}$

(*ii*) $p_{ij} > p_{kj}$

If the processing on machine M_i takes more time than the processing on machine M_k, the units finished on M_i cannot immediately be transferred to M_k. Otherwise we have to wait $p_{ij} - p_{kj} > 0$ units of time for the arrival of the next unit of product j at machine M_k. To allow a non-preemptive processing of product j on M_k, the first unit will be started on M_k $x_j(p_{ij} - p_{kj})$ units of time after its completion on machine M_i (cf. Figure 10). For the time lag $T_{ij,kj}$ we obtain

(4.2) $$T_{ij,kj} = \vartheta_{ij} + x_j p_{ij} - (x_j - 1)p_{kj} - \vartheta_{kj}$$

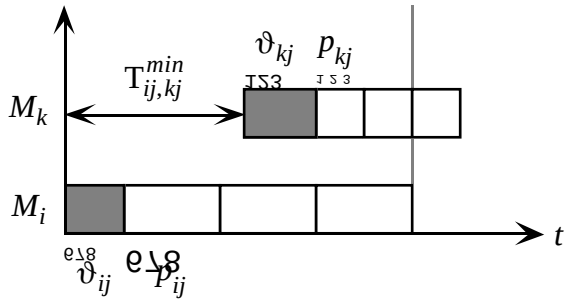


Fig. 10: Overlapping operations with $p_{ij} > p_{kj}$

A *transfer time* t_{ik}^j of product j from machine M_i to machine M_k can be taken into account by adding t_{ik}^j to the time lag:

(4.3) $$T_{ij,kj} = \begin{cases} \vartheta_{ij} + p_{ij} - \vartheta_{kj} + t_{ik}^j, & \text{if } p_{ij} \le p_{kj} \\ \vartheta_{ij} + x_j p_{ij} - (x_j - 1)p_{kj} - \vartheta_{kj} + t_{ik}^j, & \text{otherwise} \end{cases}$$

Remark 6. If $x_j = 1$, there is no difference between non-overlapping and overlapping processing. In this case, formula (4.3) provides the same value for both cases.

Automated transfer systems like AGVs (automated guided vehicles) or conveyers normally use pallets or standardized boxes (cf. Askin & Standridge, 1993). The *transportation lot size* q_{ik}^j will then be greater than one. To obtain the appropriate time lags, we have to increase $T_{ij,kj}$ by the processing time of $q_{ik}^j - 1$ units of product j on machine M_i or machine M_k, depending on whether $p_{ij} \le p_{kj}$ or $p_{ij} > p_{kj}$:

(4.4) $$T_{ij,kj} = \begin{cases} \vartheta_{ij} + q_{ik}^j p_{ij} - \vartheta_{kj} + t_{ik}^j, & \text{if } p_{ij} \le p_{kj} \\ \vartheta_{ij} + x_j p_{ij} - (x_j - q_{ik}^j)p_{kj} - \vartheta_{kj} + t_{ik}^j, & \text{otherwise} \end{cases}$$

Remark 7. If $x_j = q_{ik}^j$, there is no difference between non-overlapping and overlapping processing. In this case, formula (4.4) provides the same value for both cases.

If there is a minimum number of units $f_{ik}^j > 1$ which have to be processed on machine M_i before the first lot of product j can be passed over to the succeeding machine M_k, we have to distinguish between three cases:

(*i*) $p_{ij} \le p_{kj}$

The processing time $q_{ik}^j p_{ij}$ has to be replaced by $\max\left\{f_{ik}^j, q_{ik}^j\right\} \cdot p_{ij}$.

(*ii*) $p_{ij} > p_{kj}$ and $x_j p_{ij} - (x_j - q_{ik}^j) p_{kj} \ge f_{ik}^j p_{ij}$

Since the processing-time-based time lag $x_j p_{ij} - (x_j - q_{ik}^j) p_{kj}$ is greater than or equal to the time needed for the completion of f_{ik}^j units on machine M_i, the time lag remains unchanged.

(*iii*) $p_{ij} > p_{kj}$ and $x_j p_{ij} - (x_j - q_{ik}^j) p_{kj} < f_{ik}^j p_{ij}$

In this case (which has not been considered by Günther, 1992), the first transportation lot can be transferred to the successive machine M_k immediately after the completion of f_{ik}^j units on machine M_i.

Remark 8. It can easily be shown that in case of $p_{ij} > p_{kj}$, $x_j p_{ij} - (x_j - q_{ik}^j) p_{kj} < f_{ik}^j p_{ij}$ implies $f_{ik}^j > q_{ik}^j$.

In summary, for linear product structures, we obtain the following time lag $T_{ij,kj}$ between the start of two successive operations (i,j) and (k,j):

$$(4.5) \qquad T_{ij,kj} = \begin{cases} \vartheta_{ij} + \max\left\{f_{ik}^j, q_{ik}^j\right\} \cdot p_{ij} - \vartheta_{kj} + t_{ik}^j, \text{ if } p_{ij} \le p_{kj} \\ \vartheta_{ij} + x_j p_{ij} - (x_j - q_{ik}^j) p_{kj} - \vartheta_{kj} + t_{ik}^j, \\ \qquad \text{if } p_{ij} > p_{kj} \text{ and } x_j p_{ij} - (x_j - q_{ik}^j) p_{kj} \ge f_{ik}^j p_{ij} \\ \vartheta_{ij} + f_{ik}^j p_{ij} - \vartheta_{kj} + t_{ik}^j, \text{ otherwise} \end{cases}$$

Remark 9. If $x_j = \max\left\{f_{ik}^j, q_{ik}^j\right\}$, there is no difference between non-overlapping and overlapping processing. Then formula (4.5) provides the same time lag in any case.

(b) Convergent product structure

The structure of a product is said to be convergent if the corresponding gozinto graph represents an intree. Products of a convergent structure do not contain any repeat part. Let j be a component or subassembly which has to be built in the subassembly or end item l, respectively, and let a_{jl} denote the corresponding input coefficient. To determine the time lag $T_{ij,kl}$ between the start of setting up the last machine M_i for product j and setting up the first machine M_k for product l, we have to consider three cases in analogy to a linear product structure. We obtain

$$T_{ij,kl} = \begin{cases} \vartheta_{ij} + \max\left\{a_{jl}, f_{ik}^{j}, q_{ik}^{j}\right\} \cdot p_{ij} - \vartheta_{kl} + t_{ik}^{j}, \text{ if } a_{jl}p_{ij} \le p_{kl} \\ \vartheta_{ij} + a_{jl}x_l p_{ij} - (x_l - \frac{q_{ik}^{j}}{a_{jl}})p_{kl} - \vartheta_{kl} + t_{ik}^{j}, \\ \qquad \text{if } a_{jl}p_{ij} > p_{kl} \text{ and } a_{jl}x_l p_{ij} - (x_l - \frac{q_{ik}^{j}}{a_{jl}})p_{kl} \ge f_{ik}^{j}p_{ij} \\ \vartheta_{ij} + \max\left\{a_{jl}, f_{ik}^{j}\right\} \cdot p_{ij} - \vartheta_{kl} + t_{ik}^{j}, \text{ otherwise} \end{cases} \tag{4.6}$$

Remark 10. If $a_{jl} = 1$, (4.6) corresponds to formula (4.5).

(c) General product structure

The determination of appropriate time lags becomes more difficult if we have to deal with products which are components of more than one subassembly or final product (repeat parts).

Let, for product j, P_j be the set of all products l with $a_{jl} > 0$. Let, for product l, $\overline{P}_l$ be the set of products j with $a_{jl} > 0$. Moreover, let $j(l)$ be the job which consists of the manufacturing of $x_l a_{jl}$ units of product j needed for the assembly of the demand of product l, and let subjob $j_{r,s}(l)$ denote the processing of the r-th unit of product j built in the s-th unit of product l ($r = 1,..., a_{jl}$, $s = 1,..., x_l$). Before we are able to determine time lags in a similar manner as for a linear or convergent product structure, we have to decide on the sequence $S(j)$ in which the finished units of j have to be allotted to the units of products $l \in P_j$. Clearly, the maximal completion time of units of products $l \in P_j$ depends on the *allotment sequence* $S(j)$.

Let p_λ be the time between two successive completions of units of product λ ($\lambda \in P_j \cup \{j\}$). An estimation of p_λ is given by $\tilde{p}_\lambda := \sum_{M_i \in M_\lambda} \frac{p_{i\lambda}}{m_i}$. Let $C_{j_{r,s}(l)}$ be the completion time of subjob $j_{r,s}(l)$ given that the processing of the required units of product j is started at time 0, and let C_{l_s} be the completion time of the assembly of the s-th unit of product l, given that all required units of products $j' \in \overline{P}_l \setminus \{j\}$ have been completed at this time. Then

$$V_j(t) := \left\{ j_{r,s}(l) \middle| l \in P_j, r = 1,\ldots,a_{jl}, s = 1,\ldots,x_l, t < C_{j_{r,s}(l)} \le t + p_j \right\}$$

represents the set of jobs $j_{r,s}(l)$ which are in execution at time t, and

$$W_l(t) := \left\{ s \in \{1,\dots,x_l\} \middle| t < C_{l_s} \le t + p_l \right\}$$

is the set of units of products l which are assembled at time t.

We state the allotment sequencing problem ASP(j) for units of product j:

$$\text{ASP}(j) \quad \begin{cases} \min & \max\limits_{l \in P_j} C_{j(l)} & \\ \text{s.t.} & C_{j(l)} \ge \max\limits_{r,s} C_{j_{r,s}(l)} + p_l & (l \in P_j) \\ & \left| V_j(t) \right| \le 1 & (t = 0,\dots, p_j \sum\limits_{l \in P_j} x_l a_{jl} - 1) \\ & \left| W_l(t) \right| \le 1 & (l \in P_j, t = 0,\dots, \sum\limits_{l \in P_j} x_l (a_{jl} p_j + p_l) - 1) \\ & \min\limits_{l \in P_j} \min\limits_{r,s} C_{j_{r,s}(l)} = p_j & \\ & \text{Splitting of subjobs} & \\ & j_{r,s}(l) \text{ is not allowed} & (l \in P_j, r = 1,\dots, a_{jl}, s = 1,\dots, x_l) \end{cases}$$

ASP(j) says that the maximum completion time of all products $l \in P_j$ in which units of product j have to be built is to be minimized. The completion time of all units of product $l \in P_j$ is greater than or equal to the the completion of the last unit of product j allotted to product l plus the time interval between two successive completions of units of product l. At most one unit of product j or product l, respectively, can be processed at the same time. The processing of the first unit of product j is started at time 0. The processing of units of product j cannot be interrupted.

Remark 11. The objective function of ASP(j) has been chosen in accordance with the objective function $ST_{n+1} = \max\limits_{j=1,\dots,n} \left\{ ST_j + D_j \right\}$ of RCPSP / max.

As an example we consider some product II shown in Figure 11.

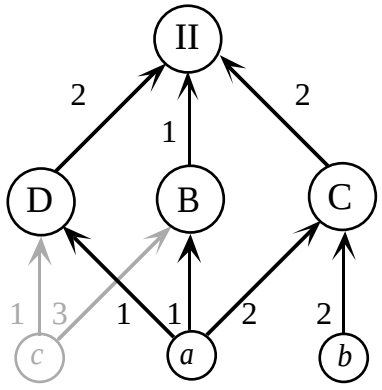


Fig. 11: Gozinto graph

For the assembly of one unit of product II, 7 units of product *a* are needed in total. If the order size of II equals one, we obtain 105 possible sequences for the allotment of the 7 units of component *a* to the units of subassemblies B, C, and D. A possible allotment sequence of *a* is *S*(*a*) = (C,D,C,B,C,D,C). With the order quantities given in Figure 4, the number of possible sequences grows up to $\frac{(16+6+4+2)!}{16!6!4!2!} \approx 4.216 \cdot 10^{11}$.

In general, we obtain

$$(4.7) \qquad \left(\sum_{l \in P_j} x_l \, a_{jl} \right)! \Big/ \prod_{l \in P_j} \left(x_l \, a_{jl} \right)!$$

possible sequences for the allotment of units of product *j* to products *l* with $a_{jl} > 0$.

In what follows, we will consider the problem ASP(*j*)/block, a special case of problem ASP(*j*), where the allotment sequence *S*(*j*) is of the form of a so-called *block structure*:

$$S(j) = \left(\underbrace{l_1, \ldots, l_1}_{a_{jl_1} x_{l_1}}, \underbrace{l_2, \ldots, l_2}_{a_{jl_2} x_{l_2}}, \ldots, \underbrace{l_\nu, \ldots, l_\nu}_{a_{jl_\nu} x_{l_\nu}} \right) \quad (\nu = |P_j|).$$

Remark 12: We obtain a formulation of ASP(*j*)/block by replacing "Splitting of subjobs $j_{r,s}(l)$ is not allowed $(l \in P_j, r = 1, \ldots, a_{jl}, s = 1, \ldots, x_l)$" with "Splitting of jobs *j*(*l*) is not allowed $(l \in P_j)$" in the formulation of ASP(*j*).

In problem ASP(*j*)/block, the number of possible sequences is reduced to ν!. The following Algorithm 2 provides an optimal solution of ASP(*j*)/block:

Algorithm 2. Allotment sequencing problem with block structure ASP(j)/block

Step 1: Initialization

Let $q_l := \max\left\{ x_l \, p_l - (x_l - 1) \, a_{jl} \, p_j ; p_l \right\}$ $(l \in P_j)$ and *S*(*j*) := ( ).

Step 2: Assignment

Put the products $l \in P_j$ in order of nonincreasing values of q_l in the sorted list $L = (l_1, l_2, \ldots, l_\nu)$. Break ties arbitrarily.

FOR λ = 1 TO $|P_j|$ DO
    FOR μ = 1 TO $x_{l_\lambda} \, a_{jl_\lambda}$ DO
        S(*j*):= *S*(*j*) ol_λ (put l_λ at the end of list *S*(*j*))
    END (* FOR *)
END (* FOR *) ❑

Remark 13. The time complexity of Algorithm 2 is $O\left(\left|P_j\right|\log\left|P_j\right|\right)$

Theorem 2. Algorithm 2 solves ASP(*j*)/block to optimality.

Sketch of proof.

A well-known result of job-shop scheduling theory states that the single-machine problem $1||L_{\max}$ (minimization of the maximum lateness of jobs *j* with given due dates δ_j) can be solved to optimality by the *EDD-rule* (Earliest Due Date first), where the *lateness* L_j of job *j* is defined as $L_j := C_j - \delta_j$, C_j being the completion time of job *j*.

By setting $q_j := \delta - \delta_j > 0$ (δ being an appropriate constant), we obtain the corresponding single-machine problem $1|q>0|C_{\max}$ (minimization of the maximal completion time of all jobs *j* with given *tails* q_j, where q_j is the time the processed units of products *j* have to stay in the system after the completion of the corresponding job).

Let *S* be the solution space of the given problem of type $1||L_{\max}$. Obviously, *S* will also constitute the solution space of the corresponding problem of type $1|q>0|C_{\max}$. Since

$$\min_{S\in S} L_{\max}(S) = \min_{S\in S}\max_j\left\{C_j(S)-\delta_j\right\} = \min_{S\in S}\max_j\left\{C_j(S)+q_j\right\}-\delta = \min_{S\in S}\left\{C_{\max}(S)\right\}-\delta,$$

each optimal sequence $S^* \in S$ for problem $1||L_{\max}$ is an optimal sequence for the corresponding problem $1|q>0|C_{\max}$, too.

In Step 1 of Algorithm 2 we set $q_l := \max\left\{x_l p_l - (x_l - 1)a_{jl}p_j ; p_l\right\}$. Since q_l represents the amount of time the $a_{jl}x_l$ units of product *j* manufactured for the demand of product *l* have to spend in the system after their completion, ASP(*j*)/block is equivalent to the corresponding problem $1||L_{\max}$, which can be solved to optimality by applying the EDD-rule. ❑

Example 1.

Consider the product structure of Figure 3. Table 1 shows the processing and assembly times of a unit of products *a*, B, C, and D, respectively.

Product	Processing / assembly time
a	$p_a = 1$
D	$p_D = 1$
B	$p_B = 4$
C	$p_C = 3$

Tab. 1: Processing and assembly times

We assume that all items of components *b* and *c* needed for the assembly of B, C, and D have been completed at time 0.

Applying Algorithm 2 to ASP(*a*) we obtain *S*(*a*) = (B,C,C,C,C,D,D) or *S*(*a*) = (C,C,C,C,B,D,D) both leading to $\max_{l\in P_a} C_{a(l)} = 9$ (cf. Figure 12).

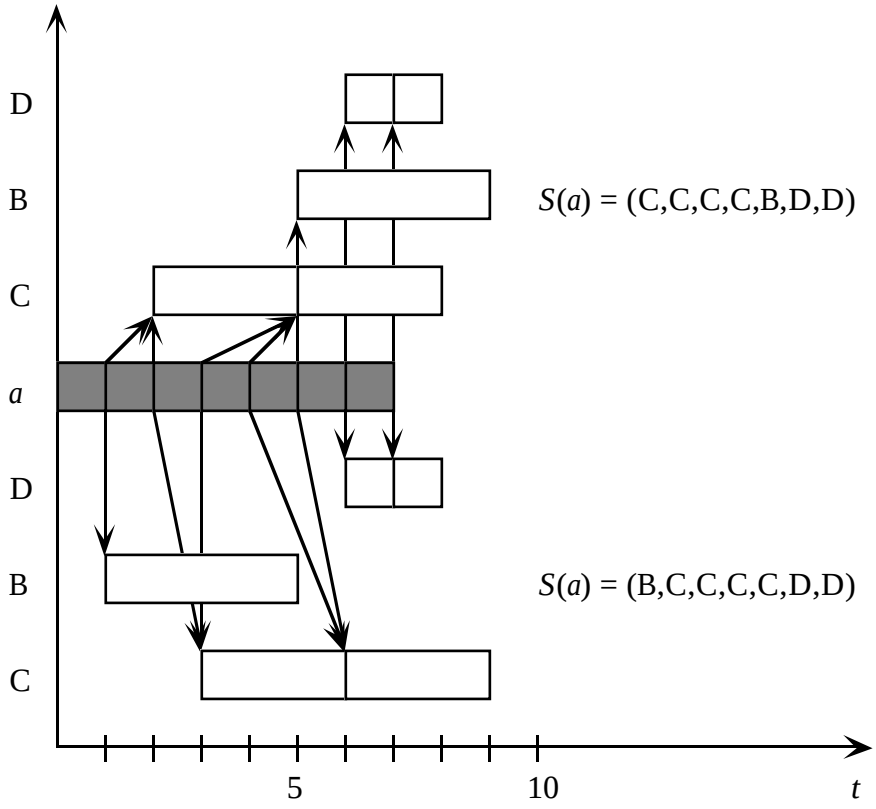


Fig. 12: Gantt chart for the allotment of units of a repeat part

As we will see later, there is a feasible solution $S(a)$ = (C,C,B,C,C,D,D) to ASP(a) with $\max_{l \in P_a} C_l$=8. Hence, the restriction to block structures may lead to suboptimal solutions for the allotment sequencing problem. Therefore, we will propose a heuristic which (approximately) solves the more general allotment sequencing problem ASP(j). In the following Algorithm 3, let $P \subseteq P_j$ be the set of subassemblies or final products for which not all required units of product j have been allotted. Products $l' \in P$ with large remaining processing time are scheduled first.

Algorithm 3. Allotment sequencing problem ASP(j)

Step 1: Initialization

$S(j) := (\,)$, $P := P_j$, $\hat{a}_{jl} := a_{jl}$ for all $l \in P_j$.

Step 2: Assignment

WHILE $P \neq \varnothing$ DO

Select the product $l' \in P$ with the largest *remaining* processing time (including the manufacturing of the required units of j):

$l' := \arg\max_{l \in P} \left\{ \max \left\{ \left((x_l - 1) a_{jl} + \hat{a}_{jl} \right) p_j + p_l ;\ \hat{a}_{jl} p_j + x_l p_l \right\} \right\}$. Break ties arbitrarily.

$S(j) := S(j) \circ l'$.

$\hat{a}_{jl'} := \hat{a}_{jl'} - 1.$
IF $\hat{a}_{jl'} = 0$ THEN
$x_{l'} := x_{l'} - 1.$
$\hat{a}_{jl'} := a_{jl'}.$
IF $x_{l'} = 0$ THEN
$P := P \setminus \{l'\}$
END (* IF *).
END (* IF *).
END (* WHILE *). ❑

Remark 14. The time complexity of Algorithm 3 is $O\left(x_j \log\left|P_j\right|\right)$.

Applying Algorithm 3 to ASP(a) in Example 1, we obtain the sequences $S(a)$ = (C,C,B,C,C,D,D) or $S(a)$ = (C,C,C,B,C,D,D). For both sequences, the assembly of product II can be started at $ST_{\mathrm{II}} = 8$, which is obviously optimal.

Once an allotment sequence $S(j)$ has been generated, we have to determine suitable time lags between the last operation of a repeat part and the first operations of the parts in which one or several units of the repeat part have to be built. Let us consider the product structure shown in Figure 13. If $S(j)$ contains the block sequence $(\underbrace{l_1 l_1 \cdots l_1}_{x_{l_1}}, \underbrace{l_2 l_2 \cdots l_2}_{x_{l_2}})$, we obtain

$$T_{ij,kl_1} = \begin{cases} \vartheta_{ij} + \max\left\{a_{jl_1}, f_{ik}^j, q_{ik}^j\right\} p_{ij} - \vartheta_{kl_1} + t_{ik}^j, \text{ if } a_{jl_1} p_{ij} \le p_{kl_1} \\ \vartheta_{ij} + a_{jl_1} x_{l_1} p_{ij} - (x_{l_1} - \frac{q_{ik}^j}{a_{jl_1}}) p_{kl_1} - \vartheta_{kl_1} + t_{ik}^j, \\ \qquad \text{if } a_{jl_1} p_{ij} > p_{kl_1} \text{ and } a_{jl_1} x_{l_1} p_{ij} - (x_{l_1} - \frac{q_{ik}^j}{a_{jl_1}}) p_{kl_1} \ge f_{ik}^j p_{ij} \\ \vartheta_{ij} + \max\left\{a_{jl_1}, f_{ik}^j\right\} p_{ij} - \vartheta_{kl_1} + t_{ik}^j, \text{ otherwise} \end{cases} \tag{4.8}$$

and

$$T_{ij,ml_2} = \begin{cases} \vartheta_{ij} + (a_{jl_1} x_{l_1} + \max\left\{a_{jl_2}, f_{im}^j, q_{im}^j\right\}) p_{ij} - \vartheta_{ml_2} + t_{im}^j, \text{ if } a_{jl_2} p_{ij} \le p_{ml_2} \\ \vartheta_{ij} + (a_{jl_1} x_{l_1} + a_{jl_2} x_{l_2}) p_{ij} - (x_{l_2} - \frac{q_{im}^j}{a_{jl_2}}) p_{ml_1} - \vartheta_{ml_1} + t_{im}^j, \\ \qquad \text{if } a_{jl_2} p_{ij} > p_{ml_2} \text{ and } a_{jl_2} x_{l_2} p_{ij} - (x_{l_2} - \frac{q_{im}^j}{a_{jl_2}}) p_{ml_2} \ge f_{im}^j p_{ij} \\ \vartheta_{ij} + (a_{jl_1} x_{l_1} + \max\left\{a_{jl_2}, f_{im}^j\right\}) p_{ij} - \vartheta_{ml_1} + t_{im}^j, \text{ otherwise} \end{cases} \tag{4.9}$$

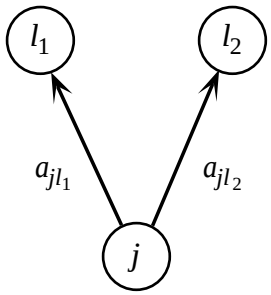


Fig. 13: Product structure

If the allotment sequence $S(j)$ is of a general form, we obtain different formulas for the time lags for almost every sequence $S(j)$. In this case, appropriate time lags can be determined by evaluating the Gantt chart (cf. Figure 12) corresponding to S(j).

4.4 Construction of a Multi-Project Network

Having determined minimal and maximal time lags between the start of successive operations, we can generate a network that corresponds to the make-to-order production scheduling problem in question. First, we will construct a separate network for each final product. These individual networks will then be put together to form a multi-project network representing the manufacturing of all ordered products including required subassemblies and components.

The following Algorithm 4 generates networks representing the manufacturing of single final products:

Algorithm 4. Construction of an acyclic A-on-N network for final product l

Replace each node j corresponding to a non-purchased product j in the gozinto graph of product l by the respective sequence of operations. To do so, each operation of job j is assigned to a node, and the sequence of operations $((i,j),...,(k,j))$ that make up job j is represented by a path $<(i,j),...,(k,j)>$ from the node belonging to the first operation (i,j) to the node belonging to the last operation (k,j). Weight the arcs with the (positive or negative) time lags between the start of the incident operations using the formulas developed in Subsections 4.3 a, b, and c. ❑

Figure 14 shows the resulting network structure for final product I given by Figure 4.

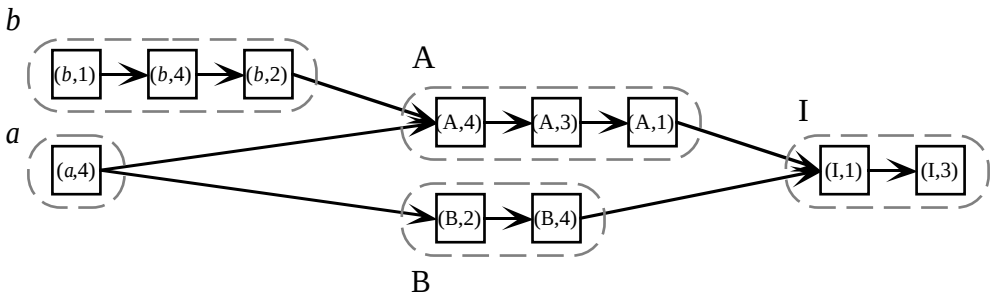


Fig. 14: Single final-product network structure

Next, we construct the multi-project network. This network will contain all single-product networks generated by Algorithm 4. Furthermore, we have to observe the deadlines of the customer orders. Upper bounds on the waiting time during the processing of products can be modelled using additional maximal time lags. Algorithm 5 describes the construction of the multi-project network.

Remark 15: In contradiction to Convention 1, a single final-product network constructed by Algorithm 4 may contain arcs with negative weights outside of cycles. In that case, dummy activities and auxiliary arcs have to be introduced as shown in Section 3 (cf. Figures 1 and 2) to obtain an A-on-N network which satisfies Convention 1. This will be done in the following Algorithm 5.

Algorithm 5: Construction of the multi-project network.

The set of nodes V of the multi-project network is the union of the node sets of all single final-product networks and the set consisting of a *supersource* α and a *supersink* ω. Thus, each product, even if built in several final products, will be represented only once.

At the beginning, the set of arcs E of the multi-project network is to be the union of the arc sets of all single final-product networks. Then we introduce arcs from the supersource α to all sources (i,j) of the individual final-product networks as well as arcs from all sinks (k,l) of the individual final-product networks to the supersink ω. The arcs emanating from α have weight 0. The weight of each arc $<(k,l),\omega>$ with final node ω equals the duration of the terminal operation (k,l).

We introduce dummy activities and auxiliary arcs as shown in Section 3 so that each backward arc with negative weight (which corresponds to a maximal time lag) belongs to a cycle.

Products j in which purchased products are directly built will be connected with α by an arc $<\alpha,(i,j)>$ with weight $r_{j'}$, where (i,j) is the first operation of job j and j' is the purchased component built in j with the maximum delivery date $r_{j'}$.

Figure 15 shows the resulting network structure for the customer orders of Figure 3.

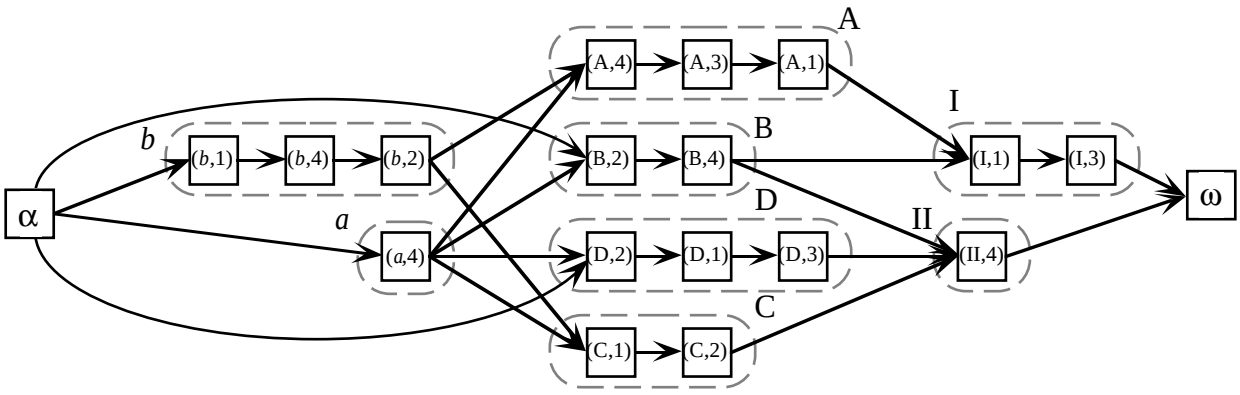


Fig. 15: Multi-project network structure

Deadlines d_j for ordered products $j \in J$ can be represented by maximum time lags $T^{max}_{\alpha,kj}$ between the supersource α and the last operation (k,j) of j. The arc from node (k,j) to node α belonging to $T^{max}_{\alpha,kj}$ has weight $-(d_j - D_{kj})$.

To avoid long waiting times between the processing of operations of one and the same product, we may establish additional time windows as follows. We add a maximal time lag $T^{max}_{ij,kj}$ between the first and the last operation of a product j, (i,j) and (k,j), respectively. By setting

$$(4.10) \quad T^{max}_{ij,kj} := (1+\varepsilon) \sum_{M_i \in M_j} D_{ij} - D_k$$

we can limit the waiting time to $\varepsilon \cdot 100\%$ of the total processing time of all units of product j. ❑

4.5 Formulation and Solution of a Problem of Type RCPSP/max

As shown in Section 3, the multi-project A-on-N network $N = <V,E; b>$ with minimal and maximal time lags generated by Algorithm 5 gives rise to time constraints of type $ST_l - ST_j \geq b_{jl}$ for all $< j,l > \in E$ where b_{jl} is the weight of arc $<j,l>$.

In addition to the time constraints, there are constraints due to scarce resources. In manufacturing, different machine types M_i can be viewed as renewable resources i (i = 1,...,K). Each operation will require a certain amount of the capacity of any machine type specified in the schedule of operations. In general, machines employed in make-to-order production can only process one part at a time. Then the capacity R_i of resource i corresponding to machine type M_i is to be equal to the number m_i of identical machines of type M_i. If the machines of type M_i are able to carry out n_i operations in parallel, we set $R_i := m_i n_i$. For the per-period requirement r_{ij} of resource i for performing operation (i,j) we have $r_{ij} := 1$.

Having determined V, E, b_{jl} for all $< j,l > \in E$, the capacities R_i of all renewable resources i = 1,...,K and the per-period requirements r_{ij} for all resources i = 1,...,K and all products $j \in J$, we can state a problem of type RCPSP/max which corresponds to the underlying make-to-order production scheduling problem. That problem can approximatively be solved by applying one of the heuristic algorithms of Brinkmann & Neumann (1994), Neumann & Zhan (1995), or Zhan (1994). A feasible schedule for RCPSP/max will allow overlapping operations and non-preemptive processing of operations, and will guarantee the on-time delivery of all ordered products.

5. Further Applications of Projects with Maximal Time Lags

In this section, we discuss further applications which require the introduction of maximal time lags. Basic concepts are summarized in Subsection 5.1. In Subsection 5.2, it will be shown how to use these concepts to model restrictions occurring in real-life project planning problems.

5.1 Basic concepts

The following Figures 16-28 describe how to introduce minimal and maximal time lags to start or to complete activities at the same point in time, to avoid waiting times, or to ensure different types of overlappings.

In general, there are several different possibilities of modelling one and the same constraint. Our goal is to introduce a minimal number of additional arcs in the underlying network. Let $A = \{1,\ldots,n\}$ be the set of the activities which are affected by the restriction to be modelled. In what follows, we denote the number of arcs required for modelling the respective restriction by $f(n)$.

(a) Starting activities at the same point in time

To start each of the activities from $A = \{1,\ldots,n\}$ at the same point in time, we have to connect the corresponding nodes by a null cycle. The orientation of the cycle can be chosen arbitrarily. Consider the case of three activities *i*, *j*, and *k* (cf. Figure 16).

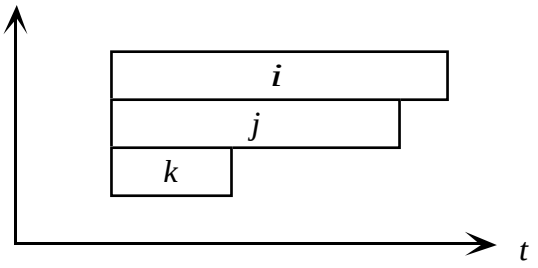


Fig. 16: Activities starting at the same point in time

Then the following equivalence holds:

$$\left.\begin{array}{ll} (i) & ST_j - ST_i \geq 0 \\ (ii) & ST_k - ST_j \geq 0 \\ (iii) & ST_i - ST_k \geq 0 \end{array}\right\} \Leftrightarrow ST_i = ST_j = ST_k.$$

(*i*), (*ii*), and (*iii*) can be represented by corresponding minimal or maximal time lags of 0 (cf. Figure 17). Since in a cycle the number of arcs equals the number of nodes, we have $f(n) = n$.

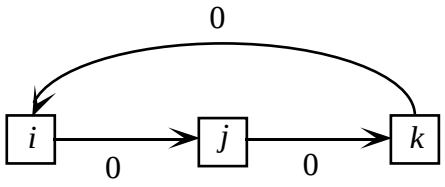


Fig. 17: Time lags ensuring coincident starts

(b) Completing activities at the same point in time

To complete each of the activities from $A = \{1,...,n\}$ at the same point in time, we can use a similar technique as above. If two activities *i* and *j* have to be terminated precisely at the same point in time, the time lag between the start of activities *i* and *j* must be equal to the difference of their durations. In case of a nonnegative difference we introduce a minimal time lag, in case of a negative difference we use a maximal time lag.

Consider the example shown in Figure 18. We have

$$\left.\begin{array}{ll}(i) & ST_j - ST_i \geq D_i - D_j \\ (ii) & ST_k - ST_j \geq D_j - D_k \\ (iii) & ST_i - ST_k \geq D_k - D_i \end{array}\right\} \Leftrightarrow \left\{\begin{array}{ll}(i) & ST_j + D_j \geq (ST_i + D_i) \\ (ii) & ST_k + D_k \geq (ST_j + D_j) \\ (iii) & ST_i + D_i \geq (ST_k + D_k) \end{array}\right\} \Leftrightarrow ST_i + D_i = ST_j + D_j = ST_k + D_k .$$

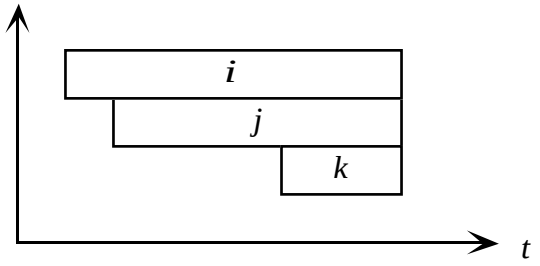


Fig. 18: Activities completed at the same point in time

Figure 19 shows the corresponding time lags. The number of required arcs is $f(n) = n$.

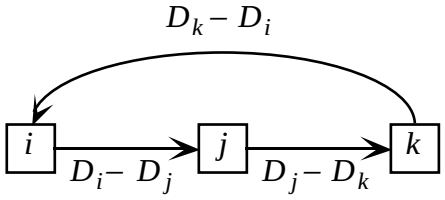


Fig. 19: Time lags ensuring coincident completions

(c) Undelayed processing of a set of activities

By an undelayed processing of a set of activities $A = \{1,...,n\}$ we mean the successive execution of all activities from A without any waiting time between two subsequent activities. Figure 20 shows the undelayed processing of three activities *i*, *j*, and *k*.

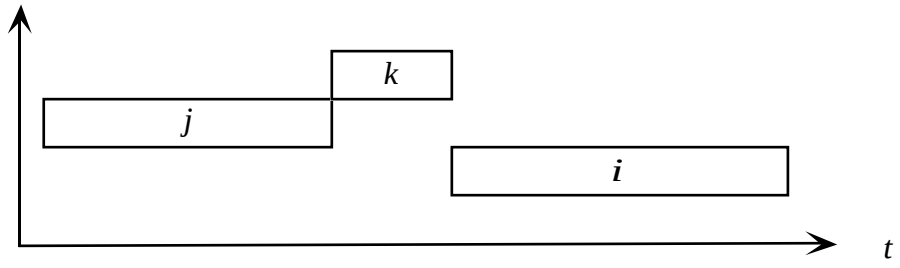


Fig. 20: Undelayed processing of activities

We assume that there are no precedence relations between the activities from A. The modelling of undelayed processing is shown in Figure 21. First, we introduce two additional nodes α and ω. Next, we add an arc from α to each activity $i \in A$ and an arc from each activity $i \in A$ to ω, where all arcs have weight 0. A fictitious resource ρ with $r_{\rho i} = 1 (i \in A)$ and $R_\rho = 1$ is introduced to avoid overlapping activities. Finally, a maximal time lag of $\sum_{i \in A} D_i$ between α and ω will ensure the undelayed processing of all activities $i \in A$. In total, $f(n) = 2n + 1$ arcs are required.

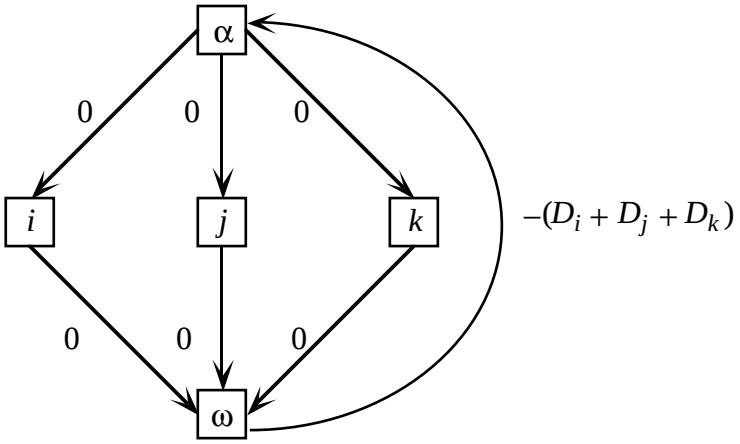


Fig. 21: Time lag ensuring undelayed processing

(d) Undelayed processing of a set of ordered activities

If the activities from $A = \{1,...,n\}$ have to be carried out in order $i_1, i_2, ..., i_n$, minimal time lags of D_ν between any two activities i_ν and $i_{\nu+1}$ $(\nu = 1,...,n-1)$ will model the prescibed order, whereas a single maximal time lag of $\sum_{i \in A \setminus \{i_n\}} D_i$ between the first activity i_1 and the last activity i_n guarantees the undelayed processing of all activities (cf. Figure 22). We have $f(n) = n$.

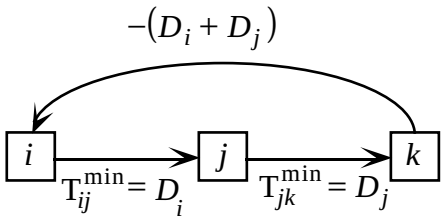


Fig. 22: Time lags ensuring undelayed processing of ordered activities

(e) Total overlapping of two activities

Let FT_i be the finish time of activity *i*.

Definition 1. Let $D_j \leq D_i$. Activity *i* is said to be *totally overlapping* activity *j* exactly if $[ST_j, FT_j] \subseteq [ST_i, FT_i]$.

Figure 23 shows three activities *i*, *j*, and *k*, where *i* is totally overlapping *j* and *j* is totally overlapping *k*. Since the relation "total overlapping" is transitive, *i* is totally overlapping *k*, too.

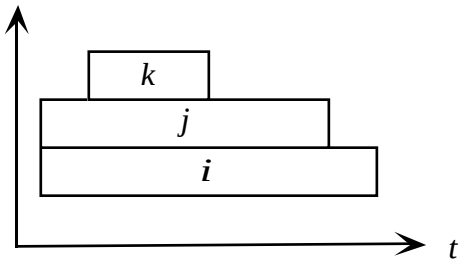


Fig. 23: Total overlappings

The modelling technique for such totally overlapping activities is based on the following equivalence:

$$\left.\begin{array}{ll}(i) & ST_j - ST_i \leq D_i - D_j \\ (ii) & ST_j - ST_i \geq 0\end{array}\right\} \Leftrightarrow \left\{\begin{array}{ll}(i) & ST_i + D_i \geq D_j + ST_j \\ (ii) & ST_j \geq ST_i\end{array}\right\} \Leftrightarrow \left\{\begin{array}{ll}(i) & FT_i \geq FT_j \\ (ii) & ST_j \geq ST_i\end{array}\right.$$

Since for each total overlapping of two activities we need one minimal and one maximal time lag, we have $f(n) = n$ (cf. Figure 24).

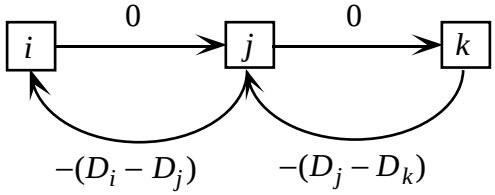


Fig. 24: Time lags ensuring total overlapping

(f) Strong partial overlapping of a set of activities

Definition 2. The activities $i \in A = \{1,\ldots,n\}$ are said to be *strongly partially overlapping* exactly if $\bigcap\limits_{i\in A} [ST_i, FT_i] \neq \emptyset$.

Figure 25 shows three activities *i, j,* and *k* which are strongly partially overlapping.

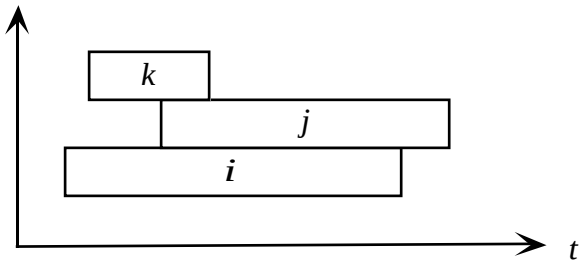


Fig. 25: Strongly partially overlapping activities

Let Δ be the prescribed minimal length of the overlapping interval $\bigcap\limits_{i\in A} [ST_i, FT_i]$ ($0 \le \Delta \le \min\limits_{i\in A} D_i$). We introduce an additional activity *l* with $D_l = 0$ (cf. Figure 26). Between each activity ν and activity *l* we add a minimal time lag of Δ and a maximal time lag of D_ν ($\nu \in A$). For the example of Figure 25 we obtain

$$\left.\begin{array}{ll} (i) & \Delta \le ST_l - ST_i \le D_i \\ (ii) & \Delta \le ST_l - ST_j \le D_j \\ (iii) & \Delta \le ST_l - ST_k \le D_k \end{array}\right\} \Rightarrow \begin{cases} (i') & ST_i - ST_j \le D_j - \Delta \quad (iv') \quad ST_j - ST_k \le D_k - \Delta \\ (ii') & ST_i - ST_k \le D_k - \Delta \quad (v') \quad ST_k - ST_i \le D_i - \Delta \\ (iii') & ST_j - ST_i \le D_i - \Delta \quad (vi') \quad ST_k - ST_j \le D_j - \Delta \end{cases}$$

Thus, any two activities $\nu_1, \nu_2 \in \{i, j, k\}$ are carried out simultaneously during at least Δ units of time, which implies the overlapping of all activities for at least Δ units of time. Figure 26 shows the corresponding network. The number of required arcs is $f(n) = 2n$.

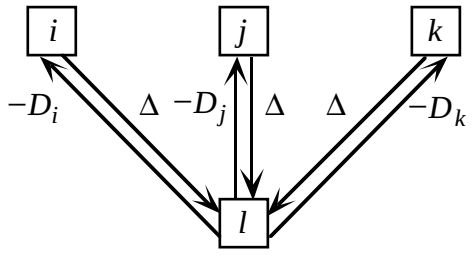


Fig. 26: Time lags ensuring strong partial overlapping

(g) Weak partial overlapping of a set of ordered activities

Definition 3. Activities $i \in A = \{1,\ldots,n\}$ are said to be *weakly partially overlapping* exactly if $\bigcup_{i\in A}[ST_i, FT_i]$ represents an interval.

Figure 27 shows three activities *i, j,* and *k* which are weakly partially overlapping.

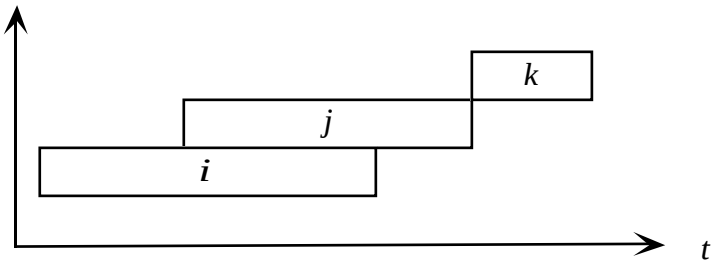


Fig. 27: Weakly overlapping activities

Weakly partially overlapping activities can be ordered such that any two successive activities overlap in time. If the activities from A have to be begun in order $i_1, i_2, \ldots, i_n$, analogously to the case of undelayed processing, minimal time lags between any two activities i_ν and $i_{\nu+1}$ $(\nu = 1,\ldots,n-1)$ will model the prescribed order. As shown in Figure 28, the weak partial overlapping can then be obtained by introducing a maximal time lag of D_ν between activities i_ν and $i_{\nu+1}$ $(\nu = 1,\ldots,n-1)$. The number of time lags used is $f(n) = 2(n-1)$.

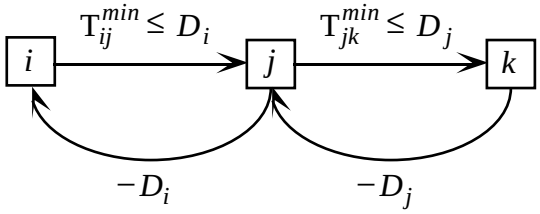


Fig. 28: Time lags ensuring weak partial overlapping

Remark 16. If *k* is totally overlapping *j* and *j* is totally overlapping *i*, this implies that *i, j, k* are strongly partially overlapping which further implies that *i, j, k* are weakly partially overlapping.

Remark 17. Any modelling technique proposed in this subsection satisfies both the first and the second part of Assumption 1 established in Section 2. That is, each maximal time lag corresponds to an arc which belongs to a cycle of nonpositive length.

5.2 Applications

(a) Chemical industry

During the production of chemical substances, some reaction products tend to be unstable. Thus, maximal time lags between chemical reactions have to be taken into account. For example, in perfume industry, almost all intermediate products cannot be stored longer than a few days.

(b) Milestones in project management

Milestones are often used in the management of large projects to impose latest finish times for some subprojects which are performed by a subcontractor. Milestones can be modelled in the same way as delivery dates in make-to-order production (cf. Section 4).

(c) Time-varying resource requirements for an activity

In problem RCPSP / max constant resource requirements per unit of time are assumed. In practice, however, the resource requirements often vary with time. To model that case for activity i, we split i into a sequence of subactivities $(i_1, i_2, \ldots, i_{n_i})$ such that each subactivity has time-constant resource requirements. We connect all subactivities by inserting arcs from i_1 to i_2, from i_2 to i_3 etc. resulting in an ordered set of activities which have to be performed without any delay. This can be done by applying the technique used in Subsection 5.1 d.

(d) Time windows for resources

In scheduling construction projects, expensive machines are often hired. Thus, they should be used only for a short period of time, a so-called time window.

Let $[r_i, d_i]$ be the time window of resource M_i and A_i be the set of activities j requiring resource M_i. To model that window, for each activity $j \in A_i$, we have to introduce an arc $<\alpha, j>$ from a supersource α to activity j representing a minimal time lag and a backward arc $<j, \alpha>$ from j to α representing a maximal time lag. The weight of arc $<\alpha, j>$ is r_i and the weight of arc $<j, \alpha>$ is $-(d_i - D_{ij})$.

Remark 18. If there exists a path from an activity j_r to an activity j_s both requiring the same resource M_i, where all arcs of the path correspond to minimal time lags, we can drop the arcs $<\alpha, j_s>$ and $<j_r, \alpha>$ due to their redundancy.

(e) Minimization of work in process (WIP) by coincident starts and completions

Consider a product structure with repeat parts which are components of several subassemblies. Suppose that these components are purchased standard parts, which will not be ordered each time they are required for the production of a subassembly. To limit the work in process (that is, the amount of raw material or intermediate products which have to be buffered), we may start several assembly operations requiring the same purchased repeat part simultaneously (such avoiding the stockpiling of the purchased components). This can be achieved by applying the technique from Subsection 5.1 a.

Instead, we may ensure that all components or subassemblies required for a given subassembly or final product, respectively, are terminated at the same point in time. In this way, no component or subassembly will have to be buffered until the start of the assembly operation. We can model that case by using the technique from Subsection 5.1 b.

(f) Due date meeting for suppliers in just-in-time systems

Just-in-time systems have been introduced by Japanese automotive industry in order to reduce inventory costs and production lead times. Just-in-time strives for coordinating activities in a way to occur as they are needed. A major element in just-in-time systems is just-in-time purchasing. The basic idea behind just-in-time purchasing is to establish agreements with suppliers to deliver small quantities of material just at the point in time when they are required (cf. Chase & Aquilano 1989). To avoid large stock of final products, we will be forced to complete orders just-in-time at the given deadline. Such a fixed completion time d_j of a job j can be ensured by introducing a minimal and a maximal time lag of $T^{min}_{\alpha,n_j} = T^{max}_{\alpha,n_j} = d_j - D_{n_j}$ between the supersource α of the corresponding multi-project network and the start time of the terminal activity n_j of job j.

(g) Reduction of fixed processing costs

The different types of overlappings introduced in Subsection 5.1 can be used to avoid fixed processing costs due to an inefficient utilization of resources. Fixed processing costs can either be utilization-independent costs for the running of a machine (for example a kiln which has high running costs regardless how many parts are in progress) or costs for the setup of a machine (for example a punch press where any replacement of the cutting tools is time consuming). In the first case, we will be interested in the total overlapping of operations in order to minimize the machine's running time (cf. Subsection 5.1 e). The setup time of a machine can be reduced by forcing activities of the same type to be processed consecutively (undelayed processing or partial overlapping, cf. Subsections 5.1 c, f, and g).

Conclusions

We have shown that maximal time lags between different activities of a project in addition to minimal ones can be modelled by a cyclic activity-on-node network. Maximal time lags play an important role in project scheduling and in production and operations management. In particular, for modelling and scheduling make-to-order production, a multi-project network can be constructed. Limited renewable resources (for example, machines), overlapping operations, and prescribed delivery dates for some parts produced can be taken into account. The resulting resource-constrained project scheduling problem can approximately be solved by heuristic procedures with reasonable computational effort. Aside from make-to-order production, we have discussed a large number of additional applications of maximal time lags in practice.

Areas of further studies are a detailed empirical analysis of the methods presented for finding appropriate time lags for general product structures and the construction of "efficient cycle structures" in the resulting multi-project network. A reduction in size of cycle structures or a partition of such structures into smaller ones, which requires an appropriate updating of arc weights, may result in a considerable reduction of the computational effort for solving the resource-constrained project scheduling problem. Moreover, the determination of milestones on an aggregate level (phase 1 of the modelling process for make-to-order production) could be improved.

References

Bartusch, M. (1983): Optimierung von Netzplänen mit Anordnungsbeziehungen bei knappen Ressourcen; *Ph.D. Thesis*, University of Aachen, Germany

Bartusch, M., Möhring, R.H., Radermacher, F.J. (1988): Scheduling Project Networks with Resource Constraints and Time Windows; *Annals of Operations Research 16*, 201-240

Brinkmann, K. (1992): Planung von deterministischen Projekten mit beschränkten Ressourcen und zeitlichen Maximalabständen; *Ph.D. Thesis*, University of Karlsruhe, Germany

Brinkmann, K., Neumann, K. (1994): Heuristic Procedures for Resource-Constrained Project Scheduling with Minimal and Maximal Time Lags: The Minimum Project-Duration and Resource-Levelling Problems; *Report WIOR-443*, Institut für Wirtschaftstheorie und Operations Research, University of Karlsruhe

Chase, R.B., Aquilano, N.J. (1990): *Production and Operations Management*; Irwin, Homewood

Christofides, N., Alvarez-Valdes, R., Tamarit, J.M. (1987): Project Scheduling with Resource Constraints: A Branch and Bound Approach; *European Journal of Operational Reserach 29*, 262-273

Demeulemeester, E.L., Herroelen, W.S. (1992): A Branch and Bound Procedure for the Multiple Resource-Constrained Project Scheduling Problem; *Management Science 38*, 1803-1818

Drexl, A., Fleischmann, B., Günther, H.-O., Stadtler, H., Tempelmeier, H. (1994): Konzeptionelle Grundlagen kapazitätsorientierter PPS-Systeme; *Zeitschrift für betriebswirtschaftliche Forschung 46*, 1022-1045

Elmaghraby, S.E. (1977): *Activity Networks*; John Wiley, New York

Elmaghraby, S.E., Kamburowski, J. (1992): The Analysis of Activity Networks under Generalized Precedence Relations; *Management Science 38*, 1245-1263

Evans, J.R., Anderson, D.R., Sweeny, D.J., Williams, T.A. (1990): *Applied Productions & Operations Management*; West Publishing, St. Paul

Franck, B., Schwindt, C. (1995): Different Resource-Constrained Project Scheduling Problems with Minimal and Maximal Time Lags - Models and Practical Applications; *Report WIOR-450*, Institut für Wirtschaftstheorie und Operations Research, University of Karlsruhe

Garey, M.R., Johnson, D.S. (1979): *Computers and Intractability*; W.H. Freedman, San Francisco

Günther, H.-O. (1992): Netzplanorientierte Auftragsterminierung bei offener Fertigung; *OR-Spektrum 14*, 229-240

Heizer, J., Render, B. (1993): *Production and Operations Management*; Allyn and Bacon, Boston

Nahmias, S. (1993): *Production and Operations Analysis*; Irwin, Homewood

Neumann, K. (1975): *Operations-Research-Verfahren, Band III*; Carl Hanser, München

Neumann, K., Lachmann, M. (1995): *Produktions- und Operations-Management*; Springer, Berlin, to appear

Neumann, K., Morlock, M. (1993): *Operations Research*; Carl Hanser, München

Neumann, K., Zhan, J. (1995): Heuristics for the Minimum Project-Duration Problem with Minimal and Maximal Time Lags under Fixed Resource Constraints; *Journal of Intelligent Manufacturing 6*, 145-154

Roy, B. (1964): *Les problèmes d'ordonnancement*; Dunod, Paris

Stinson, J.P., Davis, E.W., Khumawala, B.M. (1978): Multiple Resource-Constrained Scheduling Using Branch and Bound; *AIIE Transactions 10*, 252-259

Talbot, F.B., Patterson, J.H. (1978): An Efficient Integer Programming Algorithm with Network Cuts for Solving Resource-Constrained Scheduling Problems; *Management Science 24*, 1163-1174

Wiest, J., Levy, F. (1977): *A Management Guide to PERT/CPM*; Prentice Hall, Englewood Cliffs

Zhan, J. (1994): Heuristics for Scheduling Resource-Constrained Projects in MPM Networks; *European Journal of Operational Research 76*, 192-205

DISCUSSION PAPERS DES INSTITUTS FÜR WIRTSCHAFTSTHEORIE UND OPERATIONS RESEARCH

275. Neumann, K.: Einführung in das Operations Research I. Korrigierter Nachdruck. *Dezember 1986.*

276. Neumann, K.: Einführung in das Operations Research I I. *Juli 1986.*

277. Hofmann, H. und A. Lamatsch: Programmsystem Projektplanung - Vorstufe eines Expertensystems. *Juli 1986.*

278. Pfingsten, A.: New Concepts of Lorenz Domination and Risk Aversion. *März 1986, erscheint in:* Methods of Operations Research.

279. Pfingsten, A.: Generalized Concepts of Tax Progression and Inequality Reduction. *März 1986*.

280. Neumann, K.: Stochastic Project Networks I. *Mai 1986.*

281. Neumann, K.: Stochastic Project Networks I I. *Juli 1986.*

282. Lamatsch, A.: Einsatz des Savingverfahrens zur Wagenumlaufplanung im öffentlichen Personennahverkehr. *April 1986.*

283. Buhl, H. U.: Besprechung zu " Stochastische Optimierung bei partieller Information", Autor: P. Abel; *erschienen in:* Journal of Economics / Zeitschrift für Nationalökonomie *1986.*

284. Buhl, H. U.: Generalization and Applications of a Class of Dynamic Programming Problems, EJOR 31 (1987).

285. Bossert, W. und A. Pfingsten: The Circular and Time Reversal Tests Reconsidered in Economic Price Index Theory. *1986*; *erschienen in:* Statistical Papers 28 (1987), 271-284.

286. Neumann, K. und A. Lamatsch: Mehrgüterflüsse in Graphen zur Beschreibung des Verkehrsablaufs auf einer Strecke bei Verkehrsmischung. *Oktober 1986.*

287. Pfingsten, A.: Progressive Taxation and Redistributive Taxation: Different Labels for the Same Product? *Erschienen in:* Social Choice and Welfare 5, 1988, 235-246, *und in* Gaertner, W. und P. K. Pattanaik (Hrsg.), Distributive Justice and Inequality, Springer- Verlag, 1988, 147-158.

288. Neumann, K.: Klausuraufgaben OR I + I I mit Lösungen. Neuauflage *Mai 1989.*

289. Bossert, W.: A Note on Intermediate Inequality Indices which are Quasilinear Means. *1986.*

290. Buhl, H. U.: Theorie und Anwendungen zur Optimierung von Verschrottungserscheinungen; *erschienen in:* Isermann et al. (Hrsg.): Tagungsband der 15. DGOR Jahrestagung. OR- Proceeding of the 15. DGOR Conference 1986. Springer- Verlag, Berlin 1987.

291. Buhl, H. U. und A. Pfingsten: On the Distribution of Public Funds.

292. Fuchs-Seliger, S.: Applications of Income Compensation Functions to Social Welfare Theory.

293. Buhl, H. U.: Optimization of Scrapping Decisions: Theory and Applications; *erschienen in:* ZOR B, 1988; *erscheint in:* Radermacher, F. J. (Hrsg.), Methods of Operations Research, Athenäum- Hein, Meisenheim 1989.

294. Lamatsch, A., M. Morlock, K. Neumann und Th. Rubach: SCHEDULE - An Expert System for Scheduling. *Oktober 1986.*

295. Egle, K. und S. Fenyi: Zweistufige Disaggregation und baryzentrisches Kalkül.

296. Eichhorn, W.: On a Class of Inequality Measures; *erschienen in:* Social Choice and Welfare, 5 (1988), 171-177.

297. Lamatsch, A.: Progammbibliothek Operations Research für die Rechner HP 1000 - Siemens 7881. *Februar 1987.*

298. Buhl, H. U.: Optimale Verschrottungsentscheidungen in der Lagerhaltung; *erschienen in:* Schmalenbachs Zeitschrift für betriebswirtschaftliche Forschung, 5 (1987).

299. Buhl, H. U. und W. Bossert: More on Sufficiency Conditions for Interior Location in the Triangle Space. 1987; *erschienen in:* Journal of Regional Science (1988).

300. Buhl, H. U.: Ein finanzwissenschaftlich-methodischer Diskussionsbeitrag zur Neuordnung des Länderfinanzausgleichs in der Bundesrepublik Deutschland; *erschienen in:* Finanzarchiv, N. F. 44 (1986)3.

301. Fuchs-Seliger, S.: A note on non-inferior demand functions.

302. Eichhorn, W. und F. Stehling: Eine Bemerkung zur Verteilungsneutralität der produktivitätsorientierten Lohnpolitik. *Dezember 1986*; *erschienen in:* O. Opitz, R. Rauhut (Hrsg.): Ökonomie und Mathematik. Springer- Verlag, Berlin- Heidelberg 1987, 523-532.

303. Pfingsten, A.: Incentives to Forecast Honestly; *erschienen in:* Agency Theory, Information, and Incemtires. G. Bamberg, K. Spremann et al. (eds.). Springer- Verlag, Heidelberg 1987, 117-134.

304. Bossert, W. und A. Pfingsten: Relations between the true cost of living index and statistical price index numbers: A survey and some extensions, *1986.*

305. Fuchs-Seliger, S.: Money- Metric Utility Functions in the Theory of Revealed Preference; *erscheint in:* Mathematical Social Sciences.

306. Eichhorn, W. und M. Hellwig: Versicherungsmärkte: Theorie A. Versicherungsmärkte mit vollständiger Information; *erschienen in:* Handwörterbuch der Versicherung, HdV, herausgegeben von D. Farny, E. Helten, P. Koch und R. Schmidt, Verlag Versicherungswirtschaft, Karlsruhe 1988, 1055-1064.

307. Eichhorn, W. (editor in coorporation with W. E. Diewert, S. Fuchs- Seliger, W. Gehrig, A. Pfingsten, K. Spremann, F. Stehling and J. Voeller): Measurement in Economics. Theory and Applications of Economic Indices. Physica- Verlag, Heidelberg 1988, 830 Seiten.

308. Eichhorn, W.: Mikroelektronik - Wurzel der dritten industriellen Revolution; *erschienen in:* WÜBA- Gazette, Jubiläumsausgabe WÜBA, 150 Jahre Dienst am Kunden, 1837-1987. Heilbronn 1987, 47-51.

309. Fuchs-Seliger, S.: Measuring by Money- Metric Utility Functions.

310. Pfingsten, A.: Scaling Income Distributions.

311. Bossert, W.: The Impossibility of an Intermediate Solution in a One-dimensional Location Model: A General Result. *1987*; *erscheint in:* Regional Science and Urban Economics.

312. Weinhardt, C.: On inequality measurement when population sizes differ; *erschienen in:* Henn et al. (Hrsg.): Methods of Operations Research, 59, Athenäum Verlag (1989), S.111-124

313. Eichhorn, W. und Gleissner, W.: The Equation of Measurement; *erschienen in:* vgl. Discussion Paper No. 307, 19-27.

314. Eichhorn, W. und Gleissner W.: The Solution of Important Special Cases of the Equation of Measurement; *erschienen in:* vgl. Discussion Paper No. 307, 29-37.

315. Bossert, W. und A. Pfingsten: Intermediate Inequality: Concepts, Indices, and Welfare Implications. *1987*, *erschienen in:* Mathematical Social Sciences.

316. Fuchs-Seliger, S.: Non-inferior Demand Functions Revisted.

317. Buhl, H. U. und A. Pfingsten: 10 Gebote für Finanzausgleichsverfahren und deren Implikationen. *September 1987*.

318. Buhl, H. U.: Die " Zitronen- Kette". *Oktober 1987*.

319. Buhl, H. U.: Axiomatic Considerations in Multi- Objective Location Theory. *Oktober 1987*; *erschienen in:* European Journal of Operations Research, 1988.

320. Fuchs-Seliger, S. und M. Krtscha: An Alternative Approach to Joint Continuity in Economics.

321. Wenzelburger, D.: Vergleich verschiedener umweltpolitischer Instrumentarien anhand eines Optimiermodells.

322. Neumann, K.: Die Netzwerk- Simplexmethode zur Lösung des Umlade- und des Transportproblems. *Dezember 1987*.

323. Neumann, K.: Kürzeste Wege und Matchings. *Januar 1988*.

324. Buhl, H. U.: Besprechung zu: Anders/ Borglin " Optimality in Infinite Horizon Models"; *erschienen in:* Journal of Institutional and Theoretical Economics / Zeitschrift für die gesamte Staatswissenschaft, 1988.

325. Buhl, H. U.: Dauerarbeitslosigkeit: Wachstumstheoretische und Verteilungstheoretische Aspekte.

326. Eichhorn, W.: Risiko und Versicherung; *erschienen in:* Das Risiko und seine Akzeptanz, Hoechst-Gespräch 1988, herausgegeben von der Hoechst AG, Schütze Verlag, Bonn-Frankfurt 1989, 95-120.

327. Morlock, M.: Wissensbasierte Systeme im ORI (Heuristiken). *März 1988*.

328. Pfingsten, A.: Empirical Investigation of Inquality Concepts: A Method and First Results.

329. Neumann, K.: Das Briefträgerproblem in Graphen, Digraphen und gemischten Graphen. *März 1988*.

330. Neumann, K.: Handlungsreisendenproblem und Tourenplanung. *Juni 1988*.

331. Egle K. und S. Fenyi: Lösung des D/W Input- Output- Systems durch stochastische Inversion.

332. Pfingsten, A.: Surplus Sharing Methods.

333. Schnelle, M.: Kundenzeitschranken in der Tourenplanung. *September 1988.*

334. Zhan, J.: Kalendrierung der Terminplanung in MPM-Netzplänen. *Januar 1988.*

335. Ott, V. und J. Weber: Entscheidungsmethoden für Umwelttechnik.

336. Eichhorn, W.: Unabhängigkeit der Shephardschen Axiome; *erschienen in:* Statistik, Informatik und Ökonomie, W. Janko (Hrsg.), Springer-Verlag, Berlin-Heidelberg 1988, 49-54.

337. Fuchs-Seliger, S.: An Axiomatic Approach to Compensated Demand; *erscheint in:* Journal of Economic Theory.

338. Bossert, W.: Social Evaluation with Variable Population Size: An Alternative Concept.

339. Bossert, W.: On the Extension of Preferences over a Set to the Power Set: An Axiomatic Characterization of a Quasi- Ordering (Revised Version); *erscheint in:* Journal of Economic Theory.

340. Eichhorn, W. und U. Leopold: Logical Aspects Concerning Shephard's Axioms of Production Theory.

341. Bossert, W.: Rawlsian Welfare Orderings with Variable Population Size; *erscheint als:* " Maximin Welfare Orderings with Variable Population Size" in Social Choice and Welfare.

342. Fuchs-Seliger, S.: On the Continuity of Income Compensation Functions.

343. Bossert, W.: Generalized Gini Social Evaluation Functions and Low Income Group Aggregation; *erscheint als:* " An Axiomatization of the single-series Ginis" *in* Journal of Economic Theory.

344. Weinhardt, C.: Currency-Independence of Inequality Measures; *erschienen in:* Henn et al. (Hrsg.): Methods of Operations Research, 60, Anton Hain Verlag, (1990), S.525-536.

345. Eichhorn, W.: Vom magischen Viereck zum ökolomagischen Neuneck.

346. Buhl, H. U.: Eine Finanzanalyse des Hersteller- Leasings; *erschienen in*: Zeitschrift für Betriebswirtschaft, 4, 1989.

347. Bossert, W.: The Location of a Monopolistic Firm.

348. Bossert, W.: Population Replications and Ethical Poverty Measurement.

349. Fuchs- Seliger, S.: Non- Saturated Preferences and Compensated Demand.

350. Pfingsten, A.: Der Einsatz von monetären Anreizsystemen in der Planung.

351. Buhl, H. U.: Ein Finanzierungs- Expertensystem zur Unterstützung der Unternehmens-Strategischen Vorteile des Hersteller-Leasings; *erschienen in:* Spremann, K. (Hrsg.): Informationstechnologie und strategische Führung, Gabler, Wiesbaden 1989.

352. Pfingsten, A.: Sparten gut, alles gut? - Zur Notwendigkeit ergänzender zentraler Steuerung; *erschienen in:* Die Bank, 1989, 139-141.

353. Bossert, W. und A. Pfingsten: Nonhomothetic Preferences and Exact Price Index Numbers.

354. Hofmann, H., A. Lamatsch und M. Bücker: Programmbibliothek. *April 1988.*

355. Eichhorn, W.: Inequalities in the Theory of Economic Inequality.

356. Fuchs- Seliger, S.: Dual Models in the Theory of Demand.

357. Eichhorn, W.: Generalized Convexity in Economics: Some Examples.

358. Fuchs-Seliger, S.: Compensated and Direct Demand without Transitive and Complete Preferences.

359. Stehling, F.: Umweltschutz in der wirtschaftswissenschaftlichen Ausbildung.

360. Stehling, F.: Ökonomische Aspekte des Umweltschutzes: Ökonomie und Ökologie im Konflikt ?

361. Bücker, M. und K. Neumann: Stochastic Single-Machine Scheduling to Minimize the Weighted Expected Flow Time and Maximum Expected Lateness Subject to Precedence Constraints Given by an OR Network. *März 1989.*

362. Pfingsten, A.: Fiscal Competition and Equalization of Public Funds.

363. Geidel, J.: Richtlinien zur Dokumentation und Archivierung von Software. *März 1989*

364. Bossert, W und F. Stehling: On the Uniqueness of Cardinally Interpreted Utility Functions.

365. Eichhorn, W. und H. Funke: Prices Before and After the Vertical Merging of Firms.

366. Weinhardt, C: The Trade Off between Real Equity and Real Efficiency in Welfare Measurement Theory. Vortragsmanuskript zur Tagung der GMÖR, Ulm 1989.

367. Buhl, H. U.: Much Ado About Leasing? *Erschienen in:* Zeitschrift für Betriebswirtschaft, 8, 1989.

368. Derr, Ph.: Bericht über das Praktikum OR auf Kleinrechnern im WS 87 / 88, SS 88. *Juni 1989*

369. Eichhorn, W. und A. Vogt: Gemeinsames bei der Messung von Ungleichheit, Streuung, Risiko und Information.

370. Eichhorn, W.: Volkswirtschaftliche Auswirkungen der Mikroelektronik; *erschienen in:* Informationstechnologie und strategische Führung, Klaus Spremann und Eberhard Zur (Hrsg.), Gabler, Wiesbaden 1989, 367-377.

371. Eichhorn, W.: Equations and Inequalities in the Theory of Measurement.

372. Buhl, H. U. und G. Satzger: " Principals" und " Agents" in der Unternehmensplanung.

373. Buhl, H. U. und N. Erhard: Steuerlich linearisiertes Leasing: Kalkulation und Steuerparadoxon.

374. Geidel, J., Hoffmann D. und M. Lachmann: Teachware in Operations Research. *Dezember 1989*

375. Fuchs- Seliger, S.: Reformulation of the Theory of Demand by Compensated Demand Functions.

376. Chakravarty, S. R. und W. Eichhorn: The Optimum Size Distribution of Income.

377. Chakravarty, S. R. und W. Eichhorn: An Axiomatic Characterization of a Generalized Index of Concentration.

378. Eichhorn, W.: How not to Lie with Statistics in Regional Analysis.

379. Chakravarty, S. R.: The Optimum Size Distribution of Firms.

380. Chakravarty, S. R.: On Quasi-Orderings of Income Profiles.

381. Chakravarty, S. R.: Ethical Social Index Numbers; *erschienen als Buch:* Springer-Verlag, Berlin, Heidelberg, New York, London, Paris, Tokyo, Hong Kong, (1990).

382. Chakravarty, S. R. and B. Dutta: Migration and Welfare.

383. Chakravarty, S. R. and A. Majumder: Personal Income Distribution: Development of a New Model and Its Application to U. S. Income Data.

384. Pfingsten, A.: Profit-Based Payment Schemes in the Banking Sector.

385. Derr, Ph.: Projektplanungssoftware. *Januar 1990.*

386. Weinhardt, C.: The Central Role of Efficiency in Inequality and Welfare Measurement Theory.

387. Weinhardt, C.: The Efficiency of Price Income Situations, the Real Average Income - A Characterization -.

388. Eichhorn, W.: Das magische Neuneck. Umwelt und Sicherheit in einer Volkswirtschaft.

389. Bossert, W. und A. Pfingsten: Ordinal Utility and Economic Price Indices.

390. Bossert, W. und F. Stehling: Admissable Transformations for Interpersonally Comparable Utilities : A Rigorous Derivation.

391. Bossert, W. und F. Stehling: Social Preferences as Optimal Compromises.

392. Foulds, L. R., Hoffmann D. und K. Neumann: Stochastic Identical Parallel Machine Scheduling with OR Precedence Constraints. *March 1990.*

393. Fuchs-Seliger, S.: Non-Saturated Preferences and Compensated Demand - A Reexamination.

394. Geidel, J. und M. Lachmann: Projekt" Entscheidungsunterstützung in der Projektplanung" - Zwischenbericht. *Mai 1990.*

395. Brinkmann, K.: Bericht über das Praktikum OR auf Kleinrechnern im WS 88/89, SS 89. *Mai 1990.*

396. Weinhardt, C.: How to measure Price Progression

397. Morlock, M.: Dynamic Programming

398. Neumann, K.: Einführung in die Maschinenbelegungsplanung. *Juli 1990.*

399. Christmann, A. und W. Jorasz: Verfahren zur Aufteilung von Fertigungsgemeinkosten bei Verwendung der Bezugsgrößen Fertigungslohn und Maschinenzeiten

400. Christmann, A.: Synergetics in Kaldor`s Business Cycle Model

401. Christmann, A.: Synergetik in der Ökonomie

402. Chakravarty, S. R. and W. Eichhorn: Measurement of Income Inequalities : True versus Observed Data

403. Fuchs-Seliger, S.: A Reconsideration of Income Compensation

404. Beck, T.: Integrated Capacity and Lot Size Planning in Decentralized Production Planning. *Dezember 1990* (vergriffen).

405. Pfingsten, A. und J. Schneider: Retrieving Inequality Concepts and Progressivity Objectives From Tax Functions via Approximations

406. Geidel, J.: Praktikum " OR auf Kleinrechnern" im WS 89/90, SS 90. *Januar 1991.*

407. Lachmann, M.: Programmbibliothek Operations Research. *Januar 1991.*

408. Bücker, M., Neumann, K., Rubach, T.: Algorithms for Single-Machine Scheduling with Stochastic Outtree Precedence Relations to Minimize Expected Weighted Flow Time or Maximum Expected Lateness. *February 1991.*

409. Geidel, J., Lachmann, M.: Konzept eines modellbasierten Entscheidungsunterstützungssystems. *Februar 1991.*

410. Eichhorn, W.: Uneasy Polygons: Environment and Security Within the System of Aims of an Economy.

411. Eichhorn, W.: Produktionskorrespondenzen

412. Fuchs-Seliger: On the Evaluation of Budget Situations

413. Lachmann, M.: Begriffsgraphen. *April 1991.*

414. Bol, Morlock, Neumann, Pallaschke, Waldmann: Operations Research studieren an der Universität Karlsruhe. *September 1991.*

415. Lachmann, M.: Modelle und Verfahren in entscheidungsunterstützenden Systemen. *Dezember 1991.*

416. Bücker, M., Neumann, K.: Algorithms for Single-Machine Scheduling with Stochastic Outtree Precedence Relations to Minimize Expected Weighted Flow Time or Maximum Expected Lateness. *February 1991.*

417. Egle, K., Fenyi, S.: Eigenvalue Estimations in Input-Output and Growth Models by Monte Carlo Techniques

418. Gerhards, T.: Purchasing Power Parity and Cointegration

419. Bücker, M.: Ein Programmpaket zur Folgeplanung mit stochastischen Anordnungsbeziehungen. *Juli 1992.*

420. Fuchs- Seliger: Order Extensions and Budget Correspondences

421. Lachmann, M., Neumann, K.: A Heuristic for Multi-Product, Multi-Period, Single-Level Batch Production. *September 1992.*

422. Bücker, M.: Object Oriented Operations Research. *October 1992.*

423. Lachmann, M.: Genetische Algorithmen in der Optimierung, Bericht zum Rechnerpraktikum 1990/91. *Oktober 1992.*

424. Neumann, K.: Dynamic Programming - Basic Concepts and Applications. *October 1992.*

425. Neumann, K.: Production and Operations Management. *November 1992.*

426. Geidel, J., Lachmann, M., Präger, R. : Modellierung und Methodenauswahl in Entscheidungsunterstützungssystemen. *März 1993.*

427. Neumann, K.: Produktions- und Operations-Management I. *September 1993.*

428. Neumann, K.: Produktions- und Operations-Management II, 2. Auflage. *April 1994.*

429. Eichhorn, W., Krtscha, M.: Informationsmessung und Beziehungen zur Messung von Steuerung, Risiko, Entropie, Konzentration und Ungleichheit, *erschienen in*: Walter Frisch und Alfred Taudes (Hrsg.): Informationswirtschaft - Aktuelle Entwicklungen und Perspektiven, Physica-Verlag, Heidelberg 1993, 3-20.

430. Gerhards, T.: Strukturelle Wechselkursbeziehungen auf den Internationalen Devisenmärkten. *1993.*

431. Gerhards, T.: Zwanzig Jahre Flexible Wechselkurse - Eine Empirische Bewertung des Monetären Modells. 1993.

432. Fuchs-Seliger, S.: On Competitive Equilibria in Models of Consumer Behaviour. *1994.*

433. LS Neumann, K.: Klausuraufgaben OR I + II mit Lösungen. *September 1994.*

434. LS Neumann, K.: Klausuraufgaben POM I + II mit Lösungen. *Juni 1994.*

435. LS Neumann, K.: Klausuraufgaben Graphen und Netzwerke I + II mit Lösungen. *Juni 1994.*

436. Schneider, W.: Programmbibliothek Operations Research. *April 1994.*

437. Schwindt, C.: Vergleichende Beurteilung mehrerer Varianten der Heuristik von Lambrecht & Vanderveken zur sukzessiven Lösung des integrierten Losgrößen- und Ablaufplanungsproblems. *Mai 1994.*

438. Schnur, B.: Ein objektorientiertes Architekturmodell zur Außendienstberatung.

439. Fuchs-Seliger, S.: On Shephard's Lemma and the Continuity of Income Compensation Functions.

440. Geidel, J.: Eine graphische Modellierungssprache. *April 1994.*

441. Lachmann, M.: Ein Entscheidungsunterstützungssystem für die Standortplanung. Bericht zum Praktikum "Software-Entwicklung im Operations Research" 1992/93. *August 1994.*

442 Neumann, K., Zhan, Ji.: Heuristics for the Minimum Project-Duration Problem with Minimal and Maximal Time Lags under Fixed Resource Constraints. *August 1994.*

443 Brinkmann, K., Neumann, K.: Heuristic Procedures for Resource-Constrained Project Scheduling with Minimal and Maximal Time Lags: The Minimum Project-Duration and Resource-Levelling Problems. *November 1994.*

444 Olt, B.: Indices of Structural Changes *1995*

445 Schnur, B.: Operationalisierung von unscharfem Wissen mittels eines objektorientierten Ansatzes. *1995*

446 Schnur, B.: Realisierung eines regelbasierten Fuzzy-Systems mit der objektorientierten Entwicklungsumgebung ADS und der relationalen Datenbank MS-Access. *1995*

447 **Neumann, K., Schwindt, C.:** Projects with Minimal and Maximal Time Lags: Construction of Activity-on-Node Networks and Applications. *1995*

448 **Schneider, W.:** Job Shop Scheduling with Stochastic Precedence Constraints. *February 1995*

449 **Schwindt, C.:** ProGen / max: A new Problem Generator for Different Resource-Constrained Project Scheduling Problems with Minimal and Maximal Time Lags. *1995*

450 **Franck, B.; Schwindt, C.:** Different Resource-Constrained Project Scheduling Problems with Minimal and Maximal Time Lags - Models and Practical Applications. *1995*

451 **Redin, J.; Schneider, W.:** Programmbibliothek Operations Research - WIOR Gopher. *März 1995*

452 **Zimpelmann, M.:** Folgen des Versicherungsbinnenmarktes für die Schadenversicherung unter besonderer Berücksichtigung der Kraftfahrtversicherung. *1995*